ASIAKIRJARAKENTEIDEN SEMANTTINEN MALLINTAMINEN JA VALIDOINTI XHTML+RDFA -RAKENTEISTA

CASE SOSIAALIHUOLTO

Miika Alonen
Pro gradu -tutkielma
Tietojenkäsittelytiede

Itä-Suomen yliopiston
tietojenkäsittelytieteen laitos
Tammikuu 2012

Itä-Suomen yliopisto, Luonnontieteiden ja metsätieteiden tiedekunta
Tietojenkäsittelytieteen koulutusohjelma

MIIKA ALONEN: Asiakirjojen semanttinen mallintaminen ja validointi XHTML+RDFa-rakenteista (case sosiaalihuolto)
Pro gradu -tutkielma, 80 s., 8 liitettä
Pro gradu -tutkielman ohjaajat: FT Konstantin Hyppönen, FT Paula Leinonen
Tammikuu 2012

Avainsanat: Sosiaalihuollon asiakastietomalli, RDFa, XHTML+RDFa validointi, CCTS, RDF, RDFS, OWL, closed world validation

Tämä tutkielma perustuu Sosiaalialan tietoteknologiahankkeessa (Tikesos) tehtyyn tietoarkkitehtuurityöhön ja sosiaalihuollon asiakastietomallin kehittämiseen. Sosiaalihuollon asiakastietomalli on kuvattu CCTS-menetelmällä 148 uudelleenkäytettävästä *tietokomponenteista*, joiden avulla on yhtenäistetty 217 asiakirjarakennetta. CCTS on kansainvälinen standardi, joka ohjaa tietokomponenttien ja niihin perustuvien asiakirjarakenteiden mallintamista. Tietomäärityksten ylläpito osoittautui ongelmalliseksi, koska tietokomponenttien päivittäminen aiheutti aina muutoksia lukuisiin asiakirjarakenteisiin, XML-skeemoihin ja XSL-FO-näyttömuototiedostoihin.

Ongelman ratkaisemiseksi tutkittiin semanttisten teknologioiden käyttöä asiakirjallisen tiedon mallintamisessa sekä tietomääritysten ylläpidossa. Työn tuloksena on semanttinen mallinnusmenetelmä, jota voidaan hyödyntää asiakirjarakenteiden esittämiseen ja ylläpitoon. Mallinnustyön mahdollistamiseksi tutkielmassa on muodostettu RDF-tietomallia laajentava CCTS-metatietomalli. Tutkielmassa esitetty CCTS-metatietomalli ja asiakirjojen mallinnusmenetelmä on testattu muuntamalla Tikesos-hankkeessa mallinnetut sosiaalihuollon asiakasasiakirjat tutkielmassa esitettyyn RDF-muotoon. Sosiaalihuollon asiakastietomallin RDF-muotoa käytetään graafisten esitysmuotojen ja asiakirjojen validointipalvelun tietomallina.

Tutkielmassa käsitellään myös semanttisten mallien käyttöä asiakirjastandardeissa. Sosiaalihuollon asiakasasiakirjat voidaan muodostaa XHTML+RDFa-standardin mukaisesti yksittäisiksi rakenteisiksi asiakirjoiksi, jotka sisältävät asiakirjan rakenteen ja näyttömuodon. XHTML+RDFa-suositukseen pohjautuvan asiakirjastandardin etuna on se, että kaikki asiakirjat perustuvat samaan kansainvälisesti määriteltyyn skeemaan. Tutkielmassa on kuvattu menetelmä, jolla asiakirjojen semanttinen tietosisältö merkataan asiakirjoihin RDFa-attribuuttien avulla.

RDF-pohjaisen asiakastietomallin ja validointisääntöjen avulla voidaan validoida XHTML-asiakirjoissa RDFa-attribuuteilla merkattu tietosisältö. Tutkielmassa on kehitetty sosiaalihuollon asiakasasiakirjojen validointiin tarkoitettu XHTML+RDFa-validointipalvelu. Palvelu perustuu semanttisiin teknologioihin ja sääntöpohjaiseen validointiin. XHTML+RDFa-validointipalvelun avulla voidaan toteuttaa toiminnallisuus, joka vastaa XML-dokumentin validointia XML Schema- ja Schematron-määrityksillä.

Esipuhe

Tämä tutkielma on tehty Itä-Suomen yliopiston tietojenkäsittelytieteen laitokselle syksyllä 2011. Haluan kiittää IKE-ryhmää ja muita Sosiaalialan tietoteknologiahankkeessa työskennelleitä henkilöitä tiedon ja taidon jakamisesta. Erityiskiitos kuuluu Konstantin Hyppöselle, joka toimi tutkielman innoittajana sekä ohjaajana ja Paula Leinoselle, sekä Virpi Hotille useista mahdollisuuksista henkisen pääoman kartuttamiseen.

Kuopiossa 31.1.2012

_____

Miika Alonen

Käsitteet ja lyhenteet

Käsite	’Tiedon yksikkö, joka muodostuu käsitepiirteiden ainutkertaisesta yhdistelmästä’ [TEP11]
Asiakirjallinen tieto	’Yhteisön tehtävien hoitamisen tai henkilön toiminnan tuloksena syntyvä tai saapuva tieto’ [Lyb06]
Asiakirja	’Tallenne, jolla on oikeudellista arvoa tai joka on laadittu tai vastaanotettu jonkin yhteisön tai yksittäisen henkilön toiminnan yhteydessä tai toimintaa varten’ [TEP11]
Termi	’Erikoisalalla käytettävä yleiskäsitteen nimitys’ [TEP11]
Määritelmä	’Käsitteen kuvaus, jonka tulee erottaa käsite sen lähikäsitteistä’ [TEP11]
Semanttinen yhteentoimivuus	’Semanttinen yhteentoimivuus tarkoittaa, että tietojäjestelmä pystyy yhdistelemään eri lähteistä vastaanottamaansa tietoa ja käsittelemään sitä tavalla, jossa tietojen merkitys säilyy’ [YHT11]
Semanttinen tietomalli	Tietomalli, jossa tiedot on yksilöity ja määritelty kattavasti siten, että kaikki osapuolet ymmärtävät tiedon merkityksen ja käyttötarkoituksen samalla tavalla
Informaatio	'Datan ihmiselle tuottama mielle tai merkitys' [ATK08]
Tieto	'Asia ihmisen vastaanottamana ja ymmärtämänä informaationa, tai konkreettisessa esitysmuodossa ilmaistuna datana' [ATK08]
Tietämys	'Ihmisen oppimisen ja kokemuksen kautta hankittu, sisäistetty ja relevantiksi osoittautunut informaatio' [ATK08]
Tietokokonaisuus	'Yhdestä tai useammasta tietoryhmästä muodostuva, tietyin perustein yhteen kuuluvien tietojen joukko' [TEP11]
Tietokomponentti	'Tietokomponentit ovat semanttisia tietokokonaisuuksia, joita käytetään asiakirjallisen sisällön rakenteistamisessa. Sosiaalihuollon asiakastietomallissa tietokomponentit edustavat reaalimaailman ilmiöitä, jotka esiintyvät sosiaalihuollon asiakasasiakirjoissa. Tietokomponenttien mallintamisen perustavoitteena on varmistaa, että samantyyppiset tiedot tallennetaan asiakirjoihin yhtenäisellä tavalla. ' [STS11]

Data	’Tieto koneellisesti luettavassa, viestittävässä tai käsiteltävässä muodossa’ [ATK08]
Validointi	Objektiiviseen näyttöön perustuva varmistuminen siitä, että tiettyä käyttöä soveltamista koskevat vaatimukset on täytetty (ISO 9000)
XML	eXtensible Markup Language
XSL-FO	Extensible Stylesheet Language Formatting Objects
SGML	Standard Generalized Markup Language
W3C	World Wide Web Consortium
XHTML	eXtensible Hyper Text Markup Language
OWL	Web Ontology language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RDFa	Resource Description Framework Attributes
SPARQL	Simple Protocol and RDF Query Language
CCTS	Core Components Technical Specification
ACC	Aggregate Core Component
ASCC	Associated Aggregate Core Component
BCC	Basic Core Component
ABIE	Aggregate Business Information Entity
ASBIE	Associated Business Information Entity
NIEM	National Information Exchange Model
UN/CEFACT	United Nations Centre for Trade Facilitation and Electronic Commerce
SOAP	Simple Object Access Protocol

Sisällysluettelo

1 JOHDANTO

Asiakirjoilla on pitkä historia julkishallinnon organisaatioiden toiminnassa ja viestinvälityksessä. *Asiakirjat* toimivat viestinvälityskanavana ja todistusaineistona julkishallinnon toimijoille. Asiakirjojen avulla voidaan todentaa kaikkien osapuolien oikeudenmukainen kohtelu tai osoittaa puutteita käytetyistä toimintatavoista. Julkishallinnon toimijoiden ja yhteiskunnan oikeudenmukainen toiminta perustuu siihen, että asioiden käsittelyyn tarvittava tieto on saatavilla.

Julkisen ja kunnallisen hallinnon toimintaa on tehostettu toteuttamalla *asiakirjallisen tiedon* käsittelyyn tarkoitettuja tietoteknisiä ratkaisuja. Tietoteknisten ratkaisujen lukumäärä ja erilaiset toteutustavat kuitenkin hankaloittavat asiakirjallisen tiedon liikkuvuutta eri toimijoiden välillä. Tiedonvälitykseen eri tietojärjestelmien välillä on useita tietoteknisiä ratkaisuja, mutta oikean tietosisällön välittäminen ja sen ymmärtäminen edellyttää kuitenkin yhteisiä sopimuksia tietosisältöjen semanttisesta merkityksestä.

Tietojärjestelmien välistä yhteentoimivuutta voidaan helposti erehtyä pitämään pelkästään teknisenä ongelmana. Teknistä yhteentoimivuutta tärkeämmässä asemassa on kuitenkin yhteisesti sovittujen *käsitteiden* semanttinen merkitys ja tietojärjestelmien *semanttinen yhteentoimivuus*. Asiakirjallisen tiedon välittäminen ja automaattinen käsittely edellyttää, että kaikki tietoa käsittelevät toimijat ymmärtävät tiedon merkityksen samalla tavalla.

Sosiaalihuolto on kunnissa toteutettavaa asiakaspalvelutyötä, joka perustuu perustuslaissa määrättyihin velvollisuuksiin edistää kansalaisten hyvinvointia. Valtio rahoittaa kuntien järjestämiä palveluita, mutta vastuu sosiaalihuollon palveluiden järjestämisestä ja kustannuksista on yksinomaan kunnilla. Kuntien itsehallinto ja lainsäädäntö takaavat, että kunnilla on käytännössä vapaat kädet järjestää palveluita määrättyjen lakien puitteissa. Sosiaalihuollon palveluiden ja niihin liittyvien tietojärjestelmien hankkiminen ja kehittäminen kuntatasolla, on vaikuttanut epäsuotuisasti kansallisesti yhteentoimivien ratkaisujen kehittämiseen. Koska tietohallintoa ja tietojärjestelmäpalveluita on kehitetty ilman yhtenäistä ohjausta, on syntynyt erilaisia kunta- ja kuntakeskittymäkohtaisia ratkaisuja, jotka eivät ole yhteensopivia keskenään.

Sosiaalihuollossa käytettävät asiakirjat ja asiakirjojen tietosisällöt on yleisesti määritelty kunta- ja tietojärjestelmätoimittajakohtaisesti, joten toteutukset eivät ole yhteensopivia keskenään. Käytännössä tämä tarkoittaa sitä, että asiakirjoja ei voida siirtää sähköisesti kuntarajojen ylitse tai asiakastietojärjestelmästä toiseen. Sosiaalialan tietoteknologiahanke (Tikesos) perustettiin edistämään tietoteknologian käyttöä sosiaalihuollossa, koska on todettu, etteivät nykyisesti käytössä olevien tietojärjestelmien ongelmat poistu, ennen kuin on määritelty valtakunnallisesti yhteentoimivia tietomäärityksiä. [STM04]

Tässä tutkielmassa tehty tutkimus perustuu Tikesos-hankkeessa toteutettuun sosiaalihuollon asiakastietojen ja asiakasasiakirjojen mallinnustyöhön ja asiakirjamallinnuksessa esiin tulleiden haasteiden ratkaisemiseen. Hankkeessa on tuotettu sosiaalihuollon kansallinen asiakastietomalli ja asiakasasiakirjojen rakenteet sosiaalihuollon palvelutehtäviin ja palveluihin. Asiakastietojen ja asiakirjojen mallintamista on kuvattu luvussa 2.

Tikesos-hankkeessa kehitetyt tietomääritykset on toteutettu CCTS-menetelmää (Core Component Technical Spesification) ja kansallisia julkishallinnon suosituksia (JHS) noudattaen. Asiakirjat muodostetaan uudelleenkäytettävistä CCTS-tietokomponenteista, joiden sisältöä kehitetään yhteistyössä sosiaalihuollon ja sanastotyön asiantuntijoiden kanssa. Mallinnetut tietokomponentit määritellään taulukkomuotoisesti ja tietokomponenteista muodostetaan tietokomponenttikirjasto. Sanastotyö ja CCTS-mallinnusmenetelmä on kuvattu luvussa 2.4.

Tikesos-hankkeessa linjattiin teknisen asiakirjastandardin kehittämistä vuonna 2008, jolloin päädyttiin kehittämään sosiaalihuoltoon uusi kansallinen *XML*-pohjainen standardi. Valmiiden dokumenttistandardien ei silloin katsottu soveltuvan sellaisenaan sosiaalihuollon käyttötarkoituksiin. Tietomallin kuvaamista kokeiltiin *CCTS*-menetelmän ja XML-skeemojen avulla. XML-skeemat muodostettiin Excel-taulukoista, joilla kuvattiin asiakirjojen tietosisältöjä CCTS-menetelemän mukaisesti [HHM+09]. XML-skeemojen ja erillisten *XSL-FO*-näyttömuotojen ylläpito muodostui kuitenkin ongelmaksi, koska asiakasasiakirjojen ja asiakastietojen lukumäärä oli huomattava.

Tässä tutkielmassa on ratkaisuksi esitetty RDF-tietomalliin (Resource Description Framework) perustuvaa asiakirjojen mallinnus- ja ylläpitomenetelmää, jonka avulla voidaan muodostaa yhtenäinen semanttinen tietomalli. RDF-pohjainen asiakirjamallinnusmenetelmä hyödyntää CCTS-määrityksiä asiakirjojen rakenteiden kuvaamisessa ja RDF-kielen ominaisuuksia tietosisältöjen linkittämisessä toisiinsa. Yhtenäinen tietomal-

li vähentää päällekkäisen työn tarvetta, kun eri paikoissa käytetyt tietokomponentit on toteutettu linkittämällä tiedot yhteen. Semanttinen mallintaminen on kuvattu luvussa 3.

Muodostetulla tietomallilla voidaan hallita sosiaalihuollon tietokomponenttikirjastoa ja asiakirjoja tehokkaasti. Sosiaalihuollon asiakasasiakirjat voidaan toteuttaa XHTML+RDF-standardin avulla, joka on kuvattu luvussa 3.3.3. RDF-pohjainen tietomalli mahdollistaa skeemojen automaattisen generoinnin ja asiakirjojen sääntöpohjaisen validoinnin sosiaalihuollon asiakastietomallin avulla. Asiakirjojen validointi on kuvattu luvussa 4.

2 ASIAKIRJALLISEN TIEDON MALLINNUS

Asiakirjallisella tiedolla tarkoitetaan jonkin tietyn asian hoitamisen tuloksena syntyvää tietoa. Asiakirjallinen tieto syntyy erilaisissa toimintaprosesseissa, jotka on suunniteltu kyseisen asian hoitamiseksi. Asiakirjallinen tieto muodostuu prosessien erilaisissa vaiheissa, kuten tiedon kirjaaminen, laatiminen, rekisteröinti tai arkistointi. Asiakirjallisesta tiedosta muodostetaan asiakirjoja, jotka toimivat todisteina tehdystä työstä. Asiakirjalla tarkoitetaan yleisesti mitä tahansa informaatiota, joka on tuotettu johonkin tiettyyn tarkoitukseen ja on ymmärrettävissä tietyn menetelmän mukaisesti. Vaikka asiakirjoja tuotetaan ja käsitellään monella eri tavalla, asiakirjallisen tiedon käyttötarkoitus säilyy samana. [Lyb06]

Asiakirjallisen tiedon mallintamisella tarkoitetaan toimintatapoja ja standardeja, joilla asiakirjojen tietosisältöjä ja rakenteita mallinnetaan. Asiakirjojen mallintamiseen ja viestinvälitykseen on kehitetty useita kansainvälisiä menetelmiä. Asiakirjallisen tiedon mallintamismenetelmiä on kuvattu luvussa 2.5.

2.1 Asiakirjat sosiaalihuollossa

Sosiaalihuollon asiakirjat kuvaavat ihmisten sosiaalista hyvinvointia ja siihen vaikuttavia asioita. Sosiaalihuollon palveluissa käytettävät asiakirjat eroavat tietosisällöiltään ja käyttötarkoituksiltaan terveydenhuollossa ja yleisesti liiketoiminnassa käytettävistä asiakirjoista. Sosiaalihuollon asiakirjoihin talletetaan paljon vapaamuotoista kuvausta asiakkaan elämäntilanteesta. Vapaamuotoisten kuvausten katsotaan olevan tärkeää taustatietoa, jota sosiaalityöntekijä tarvitsee työssään asiakkaan palvelemiseen ja asiakasta koskevien päätösten tekemiseen. [HHK+08]

Sosiaalihuolto on kunnallista viranomaistoimintaa, jota ohjataan laeilla ja määräyksillä. Yleisiä sosiaalihuoltoon säädettyjä lakeja ovat sosiaalihuoltolaki (SHL 1982) ja sosiaalihuollon asiakaslaki (ASL 2000). Sosiaalihuolto voidaan jakaa 21 palvelutehtävään, kuten esimerkiksi lastensuojelu ja päihdehuolto [SPL11]. Yleisten sosiaalihuoltoa koskevien lakien lisäksi palvelutehtävien tuottamia palveluja ohjataan palvelutehtäväkohtaisilla erityislaeilla. Lait ohjaavat ja velvoittavat kuntia järjestämään sosiaalipalveluita palvelutehtävittäin ja asiakirjat toimivat muun muassa juridisena todisteena asiakkaalle annetuista palvelusta [HHL+10].

Sosiaalihuollossa kerätään ja tuotetaan paljon asiakastietoa. Asiakastietoa kerätään asiakkaan palvelutarpeen selvittämiseksi, sosiaalipalveluiden järjestämiseksi ja palveluihin liittyvien toimenpiteiden toteuttamiseksi. Asiakastiedoista koostetaan asiakirjoja, jotka talletetaan asiakastietojärjestelmiin. Asiakirjojen laatiminen on sosiaalihuollon asiakkaita palvelevien ammattihenkilöiden vastuulla. Asiakirjat toimivat sosiaalihuollon ammattihenkilöiden työvälineinä, jotka mahdollistavat asiakkaalle suunnatun sosiaalihuollon toiminnan suunnittelun, toteuttamisen ja seurannan. Asiakastyön dokumentointi tekee tehdyn työn näkyväksi ja toimii kommunikointivälineenä asiakkaan sekä sosiaalihuollon ammattihenkilöiden välillä. [LKP+11]

Henkilötietolain mukaan asiakastieto on tallennettava sosiaalipalvelukohtaisiin henkilörekistereihin, jotka on perustettu tiettyä käyttötarkoitusta varten. Asiakastietojen käyttö muuhun käyttötarkoitukseen edellyttää asiakkaan suostumusta tai muuta lakiin perustuvaa oikeutusta. Sosiaalihuollon viranomaisilla on laaja tiedonsaantioikeus työssä tarvittaviin asiakastietoihin. [LKP+11]

Sosiaalihuollon tietoteknologiahankkeessa (Tikesos) on kehitetty kansallisia tietomäärityksiä, jotka mahdollistavat sosiaalihuollon asiakastietojen kansallisen arkistoinnin ja tiedonvälityksen kuntien ja asiakastietojärjestelmien välillä. Tikesos-hankkeessa kehitetty asiakastietomalli on kuvattu luvussa 3.2.

2.2 Sanastotyö

Eri aloilla käytetään yleiskielen lisäksi erikoissanastoja, joiden avulla viestitään työyhteisössä. Sanastotyötä tarvitaan pitämään käytetty sanasto mahdollisimman yksiselitteisenä. Sanastotyöllä tarkoitetaan prosessia, jonka päämääränä on tutkittavan kohdealueen viestinnän tehostaminen. Sanastotyö sisältää *termien* yhtenäistämistä, määrittelyä, luokittelua ja standardointia [TSK11]. Sanastoilla on suuri merkitys yhtenäisten toimintatapojen ja tietojärjestelmien yhteentoimivuuden kannalta.

Sanastotyö sosiaalihuollossa on tärkeää, koska sosiaalihuollon sanasto on hyvin heterogeenistä. Sosiaalihuollon käsitteet on perinteisesti määritelty kuntatasolla, yksittäisten tietojärjestelmien näkökulmasta. Samalla sanalla voi olla useita merkityksiä eri kunnissa ja kuntien tarjoamissa palveluissa. Ilman sanastotyötä määritellyt käsitteet eivät ole suoraan ymmärrettävissä tai uudelleenkäytettävissä muissa tietojärjestelmissä. [SAN08]

Sanastotyön merkitys korostuu suunniteltaessa tiedonsiirtoa tietojärjestelmien välillä. Tiedonsiirtoa ja tietojen arkistointia suunniteltaessa on määriteltävä rajapinnat, jotka ovat kaikkien osapuolien käytettävissä. Rajapintoja määriteltäessä on tärkeää, että kaikki osapuolet ymmärtävät siirrettävien tietojen merkitykset samalla tavalla. Tätä kutsutaan semanttiseksi yhteentoimivuudeksi. Sanastotyön ja teknisen määrittelytyön yhteensovittaminen on haastavaa mutta tärkeää tietojärjestelmien semanttisen yhteentoimivuuden kannalta.

Julkisen hallinnon tietohallinnon neuvottelukunta (JUHTA) määrittelee JHS-suosituksia, joiden yhtenä tarkoituksena on edistää tietojärjestelmien yhteentoimivuutta. JHS 175 kuvaa julkisen hallinnon sanastotyöprosessin, joka käsittelee tietojärjestelmien yhteentoimivuutta tukevien sanastojen määrittelyä. Prosessin avulla määritellään kansallisesti ja kohdealuekohtaisesti yhtenäisiä käsitteitä ja määritellään käsitteille tekninen sanomarakenne. Sanaston ja teknisten sanomarakenteiden välinen yhteys mahdollistaa tietojärjestelmien välisen semanttisen yhteentoimivuuden. [JHS10]

2.3 **Koodistot**

Sosiaalihuollossa käytetään usein yhteisiä koodistoja luokittelemaan erilaisia sosiaalihuollon ilmiöitä. Yhteisesti määriteltyjen luokitusten avulla on mahdollista saada luotettavaa tilastotietoa. Koodistoja tarvitaan esimerkiksi eri ilmiöiden ja asioiden välisten riippuvuussuhteiden tutkimiseen.

Suomessa sosiaali- ja terveysalan koodistojen ylläpidosta vastaa THL (Terveyden ja hyvinvoinnin laitos). THL:n ylläpitämän koodistopalvelun tarkoituksena on ylläpitää ja jakaa kansallisesti yhteiset sosiaali- ja terveysalan tarvitsemat koodistot ja luokitukset. Koodistopalvelu tukee sosiaali- ja terveydenhuollon asiakirjojen tietosisältöjen yhtenäistämistä yhteisillä koodistoilla. Koodit yksilöidään kansainvälisen OID (ISO/IEC 8824)-standardin mukaisesti. [HLL+08]

Luokituksen käyttöönotto koodistopalvelussa edellyttää koodistoille määritellyn valmistelu- ja hyväksymisprosessin läpikäyntiä. Luokituksen hyväksymisen edellytyksenä on laadukas sisältö ja riittävä lisäarvo tarpeeksi suurelle kohderyhmälle. Edellytyksenä ovat myös uskottavat suunnitelmat koodiston käyttöönotosta ja ylläpidosta. [HLL+08]

Sosiaalihuollon koodistoja on kehitetty Tikesos-hankkeessa. Koodistoja käytetään asiakirjasisällön luokittelemiseen ja arvojoukkojen rajaamiseen. Koodistojen vieminen kansalliseen koodistopalveluun on tärkeää, koska koodistojen avulla yhtenäistetään kansallisesti sosiaalihuollon prosesseissa ja asiakirjoissa käytettäviä arvoja. Sosiaalihuollon asiakirjoissa on hyödynnetty myös terveydenhuollossa määriteltyjä koodistoja. [STS08, SHT09]

2.4 Rakenteiset asiakirjat

Rakenteisella asiakirjalla tarkoitetaan yleensä sähköistä asiakirjaa, jossa on koneellisesti käsiteltävissä oleva tekstipohjainen rakenne. Tekstipohjainen rakenne voi perustua esimerkiksi Unicode-merkistöstandardiin [UNC07], joka mahdollistaa asiakirjojen laitteisto- ja ohjelmistoriippumattoman tiedonkäsittelyn.

Rakenteiset asiakirjat muodostetaan elementeistä, jotka ilmaistaan rakenteiden kuvaamiseen varatuilla merkkijonoilla. Elementit voivat muodostaa hierarkkisia rakenteita, joiden avulla sisältöä voidaan ryhmitellä. Rakenteisten asiakirjojen hierarkkisuus mahdollistaa pelkästään tekstimuotoista asiakirjaa yksinkertaisemman ja tehokkaamman koneellisen käsittelyn [UNC07]. Rakenteen kuvaamisen lisäksi rakenteisen asiakirjan elementeillä voidaan kuvata elementin sisällön käyttötarkoitusta. Sisältöä kuvaavaa tietoa kutsutaan metatiedoksi. Metatiedoilla rikastettuja tietosisältöjä pystytään käsittelemään automaattisesti elementtien merkityksen ollessa selvillä. Rakenteisen asiakirjan elementit voivat kuvata esimerkiksi tietosisällön muotoilua. Muotoilun toteuttaminen kuitenkin edellyttää, että rakenteista asiakirjaa käsittelevä järjestelmä tunnistaa miten elementin tietosisältö tulisi muotoilla.

Rakenteisten asiakirjojen kuvaamiseen on kehitetty ISO-standardi SGML (Standard Generic Markup Language) [SGM86]. SGML-kieltä on pidetty kuitenkin sellaisenaan liian monimutkaisena asiakirjojen rakenteiden kuvaamiseen. Nykyisin käytetään SGML-standardiin pohjautuvia W3C:n suosituksia, kuten HTML (Hypertext Markup Language) ja XML (Extensible Markup Language), jotka kehitettiin erityisesti Internetin tarpeisiin, helpottamaan rakenteisten asiakirjojen määrittelyä, käsittelyä ja tiedonsiirtoa [XML08].

Tässä tutkielmassa ei kuvata rakenteisten merkkauskielten kielioppia, kuten HTML ja XHTML-kielissä käytettyjä elementtejä. Webin julkaisukielien yleistuntemus on kui-

tenkin edellytyksenä tutkielmassa esitettyjen menetelmien ymmärtämiseksi. Tutkielmassa käytettyjen XHTML-elementtien merkitykset on määritelty suosituksessa [XHT02].

2.4.1 HTML

HTML (HyperText Markup Language) on yleisnimitys webin julkaisukielelle. HTML-kielen avulla voidaan julkaista dokumentteja, jotka sisältävät tekstiä, taulukoita, kuvia sekä muuta vastaavaa informaatiota. Internetissä julkaistut dokumentit ovat luettavissa WWW (World Wide Web) -verkossa ympäri maailmaa. [HTM99]

Alun perin HTML on Tim Berners-Leen 1990-luvun alussa kehittämä kieli tutkijoiden viestinvälitykseen. Kielen kehitystä ohjaa nykyään vuonna 1994 perustettu W3C-organisaatio. HTML on webin joustavimpia julkaisukieliä, ja se toimii hyvin erilaisissa selaimissa ja käyttöjärjestelmissä. HTML-kielen rakennemäärittely mahdollistaa osittain puutteellisen dokumentin rakenteen. HTML-dokumentteja esittävät järjestelmät täydentävät puuttuvia elementtejä, ja pyrkivät esittämään myös virheelliset HTML-dokumentit mahdollisimman hyvin. [HTM99]

HTML-kieli kuvaa elementtejä lähinnä sisällön muotoiluun, joten pelkällä HTML-kielellä ei voida kuvailla tietosisällön semantiikkaa. Semantiikan kuvaamiseen on kehitetty muita W3C:n suosituksia, kuten luvussa 3.1.4 esitetty RDFa.

2.4.2 XHTML

HTML-kielestä on kehitetty XML-pohjainen merkkauskieli XHTML (eXtensible Hypertext Markup Language). XHTML on HTML-kielen kehittyneempi versio, jonka avulla web-sivustoja voidaan käsitellä tehokkaammin. Automaattinen prosessointi on mahdollista XML-kielen käsittelyyn kehitettyjen käsittelykielien ja ohjelmistojen avulla, koska XHTML edellyttää XML-kielen mukaisia muodostussääntöjä [XHT02]. XHTML mahdollistaa myös uusien rakenteiden määrittelemisen.

Uusien rakenteiden määrittely on mahdollista XHTML-modularisaation avulla. Erilliset moduulit mahdollistavat XHTML-rakenteiden supistamisen ja laajentamisen käyttöympäristön vaatimusten ja rajoitusten mukaisesti [XHM01]. Tutkielmassa on esitetty sosiaalihuollon asiakirjojen XHTML+RDFa-muoto (Luku 3.2.1), joka perustuu W3C:n Module-based XHTML -modularisaatioon.

2.4.3 XML

XML (Extensible Markup Language) on W3C:n kehittämä tekstipohjainen merkkauskieli. XML-kieli mahdollistaa omien rakenteiden ja elementtien merkitysten määrittelyn. XML-elementti muodostuu elementin aloituksesta `<esimerkki>` ja vastaavasta elementin lopetuksesta `</esimerkki>`. Elementille määriteltävä sisältö kuvataan elementin sisällä, esimerkiksi: `<esimerkki>tietosisältö</esimerkki>`. Elementti voi sisältää myös sisäkkäisiä vastaavasti muodostettuja elementtejä. Elementteihin voidaan liittää attribuutteja, esimerkiksi: `<esimerkki attribuutti="sisältö"/>`. Attribuuttien avulla voidaan tarkentaa elementtejä ja elementtien sisältöjä. XML-dokumentti muodostetaan elementeistä seuraavien pääperiaatteiden mukaisesti [XML08]:

1. XML-dokumentin tulee muodostaa validi puurakenne
2. Kaikkien elementtien tulee sijaita juuri-elementin alla
3. Jokaisen elementin täytyy olla avattu ja suljettu
4. Jokaisen elementin täytyy muodostua hyväksytyistä merkeistä
5. Attribuuteilla täytyy olla yksilölliset nimet elementtien sisällä

XML-dokumentti on hyvin muodostettu (well-formed) sen noudattaessa näitä periaatteita. Dokumenttien elementeille ja niistä muodostuville rakenteille voidaan kuitenkin määritellä tarkempia vaatimuksia. XML-skeemojen avulla voidaan varmistaa rakenteiden käyttö ja elementtien nimeäminen rakennemäärityksissä kuvatulla tavalla. XML-dokumenttien rakenteita voidaan kuvata ja validoida DTD- ja skeemarakennemäärityksillä. DTD on XML-dokumenttien kuvauskieli, jossa määritellään syntaksi ja formaalit säännöt XML-rakenteiden kuvaamiseen ja validointiin. [XML08]

XML Schema on W3C:n standardoima skeemakieli tietorakenteiden määrittelemiseen. XML-skeemat ovat XML-kielellä muodostettuja dokumentteja, joilla voidaan muodostaa DTD-rajoituksia tarkempia rajoitteita rakenteiden tietosisältöihin. Koska XML-skeemat ovat XML-muodossa, niitä voidaan käsitellä samoilla menetelmillä kuin muitakin XML-dokumentteja. [XSD04]

2.4.3.1 XML-dokumenttien käsittely

XML-pohjaisen tiedon käsittelyyn on kehitetty useita ohjelmistokirjastoja ja käsittelykieliä. Ohjelmalliseen käsittelyyn on kehitetty ohjelmistorajapinnat DOM (Document Object Model) [DOM11] ja SAX (Simple API for XML) [SAX11], jotka ovat alusta- ja kieliriippumattomia menetelmiä XML-dokumenttien käsittelyyn. Ohjelmistorajapinnoista on muodostettu toteutuksia eri ohjelmointikielille, joilla voidaan lukea ja muokata XML-dokumenttien sisältöä ja rakennetta.

XML-dokumenttien käsittely muunnostiedostojen avulla on kuitenkin monesti yksinkertaisempaa ohjelmalliseen käsittelyyn verrattuna. Muunnoskielet, kuten XSLT (XSL Transformations) [XSL07] ja XQuery [XQR10], mahdollistavat ohjelmointikielestä riippumattoman tavan käsitellä XML-dokumentteja. Muunnoskieliä käytetään yhdessä XPath (XML Path Language) -kielen kanssa, jonka avulla voidaan kohdistaa kyselyitä tai muokkausoperaatioita elementteihin XML-dokumentin puurakenteissa.

Tässä luvussa on kuvattu tarkemmin XSLT-muunnoskieli, koska tutkielmassa tarkasteltu sosiaalihuollon asiakastietomalli on muodostettu XSLT-muunnostiedostojen avulla. XSLT on W3C:n suositus, jonka avulla XML-dokumentteja voidaan käsitellä ja muuttaa eri muotoihin. XML-dokumentin muunnokset suoritetaan XSLT-prosessorin avulla, jolle annetaan yleensä syötteenä yksi lähdedokumentti ja XSLT-muunnostiedosto. Muunnostiedostossa kuvataan käsiteltävät rakenteet ja muunnossäännöt, joiden perusteella lähdedokumentin elementtejä käsitellään. Prosessori muodostaa XSLT-muunnostiedostossa määriteltyjen muunnossääntöjen pohjalta yhden tai useamman tulosdokumentin [XSL07].

XSLT sisältää laajan joukon menetelmiä, joilla XML-dokumenttien rakennetta ja sisältöä voidaan muokata. XML-dokumenttien sisältöä voidaan käsitellä XSLT-kielellä esimerkiksi käymällä tietyn nimiset elementit läpi yksi kerrallaan, tai jakamalla ensin dokumentti pienempiin osiin rekursiivisien kutsujen avulla. XSLT mahdollistaa myös ulkopuolisien XML-tiedostojen kutsumisen XSLT-muunnostiedoston prosessoinnin aikana. XSLT-muunnoksia on käytetty tässä tutkielmassa RDF-tietomallien muodostamiseen XML-pohjaisista tietokomponenttikuvauksista. XSLT-muunnostiedostoja on käytetty myös yhdessä SPARQL-kyselyiden kanssa (Luku 3.1.5).

2.5 Tietokokonaisuuksien mallintaminen

Semanttisen yhteentoimivuuden saavuttaminen edellyttää yhteisesti sovittuja menetelmiä ja sopimuksia tietojen mallinnustavasta ja asiakirjastandardeista. Asiakirjoja voidaan mallintaa käyttämällä useita eri standardeja ja suosituksia. Tunnetuimpia asiakirjojen mallintamiseen käytettyjä mallinnusmenetelmiä ovat NIEM ja CCTS.

NIEM (The National Information Exchange Model) on yhdysvaltalainen hanke, jonka tavoitteena on parantaa yhteentoimivuutta eri toimijoiden välillä. NIEM kehittää menetelmiä ja työkaluja, joiden avulla voidaan kehittää yhteentoimivia tietomalleja. NIEM-tietomalli perustuu teknologiariippumattomien tietokokonaisuuksien mallintamiseen, joiden avulla voidaan saavuttaa yhteentoimivuus eri tietomallien välillä. NIEM-mallissa määritellään joukko ydintietokomponentteja, joita voidaan tarvittaessa laajentaa toimialakohtaisesti. [NIE07]

Core Component Technical Specification (CCTS) on United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) -organisaation kehittämä menetelmä tietokomponenttien ja asiakirjarakenteiden mallintamiseen. CCTS-menetelmä mahdollistaa mallinnettujen tietokomponenttien uudelleenkäytön ja viestinvälityksen eri tahojen välillä [UNC09].

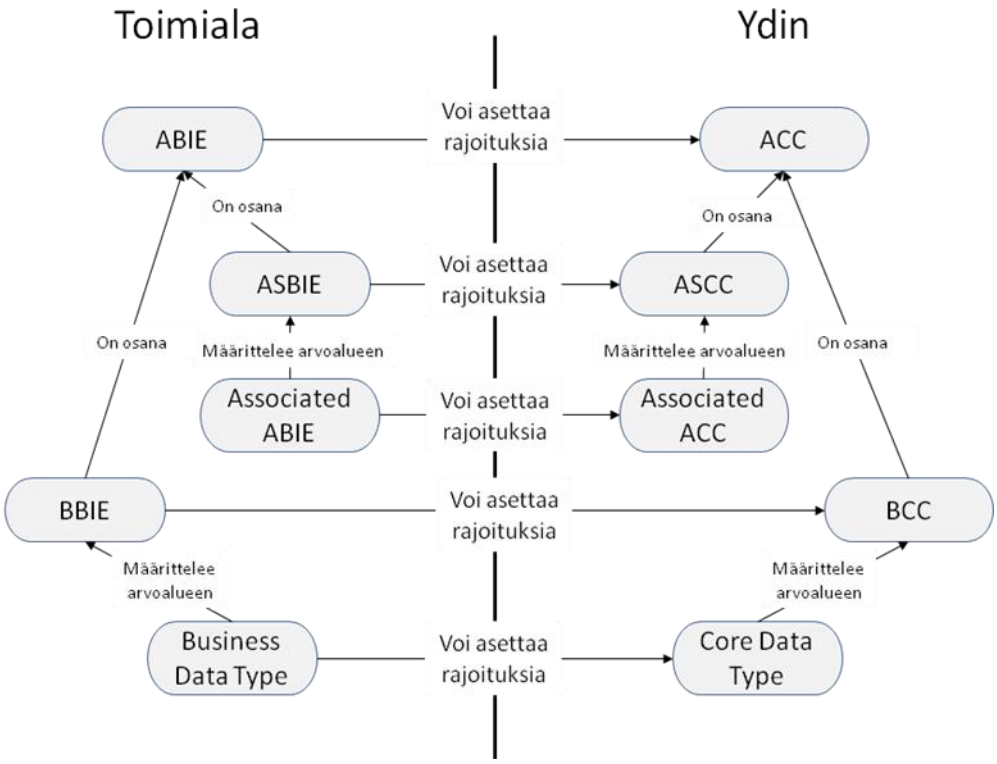
CCTS on NIEM-mallin tavoin teknologiariippumaton mallinnusmenetelmä. Molempien tietomallinnusmenetelmien käyttötarkoitus on pääperiaatteiltaan sama, kuitenkin mallien tekninen toteutus eroaa toisistaan. CCTS-mallinnusmenetelmä valittiin käyttöön sosiaalihuollossa, koska sitä sovelletaan laajasti Euroopassa [LAU10]. Kaikilla tietokokonaisuuksien mallinnusmenetelmillä on kuitenkin yleisesti ottaen yhteinen päämäärä, semanttinen yhteentoimivuus yhteisesti sovittujen tietomäärityksien avulla.

2.5.1 Core Component Technical Specification

CCTS-määrityksen keskeisin käsite on ydinkomponentti (Core Component, CC), joka toimii tietosisältöjä kokoavana tekijänä. Ydinkomponentit ovat tarkoin määriteltyjä uudelleenkäytettäviä rakennuspalikoita, joita voidaan käyttää tiedon mallintamiseen ja viestinvälitykseen [UNC09].

Ydinkomponentit toimivat yläkäsitteinä toimialakohtaisille tietokomponenteille (Business Information Entities, BIEs), joille määritellään toimialakohtainen konteksti.

Ydinkomponentin nimeen lisätään toimialakohtainen määretermi, jolla rajoitetaan ydinkomponentin käyttöä tiettyyn toimialakohtaiseen tarkoitukseen. Ydinkomponenttien ja toimialakohtaisien tietokomponenttien väliset suhteet on esitetty kuvassa 1.



Kuva 1: Toimialakohtaisten tietokomponenttien ja ydinkomponenttien välinen suhde. (Mukautettu lähteestä [CCT09])

2.5.1.1 Ydintietokomponentit

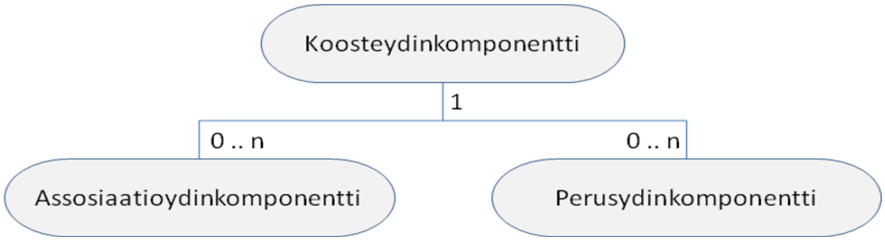
Ydinkomponentit voidaan jakaa kolmeen pääkategoriaan [UNC09]:

- Koosteydinkomponentti (Aggregate Core Component, ACC)
- Perusydinkomponentti (Basic Core Component, BCC)
- Assosiaatioydinkomponentti (Association Core Component, ASCC)

Ydinkomponenteiksi tai ydinkomponenttien osiksi voidaan katsoa kuuluvan myös assosiaatio- (ASCC Property) ja perusydinkomponenttien (BCC Property) uudelleenkäytettävät ominaisuudet sekä perusydinkomponenttien tietotyypit. [UNC09]

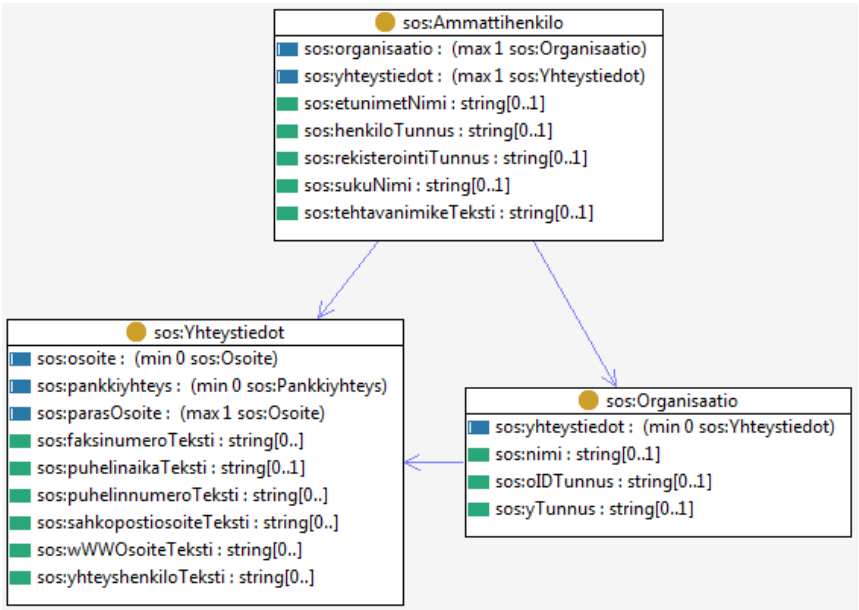
Koosteydinkomponentti on kokoelma toisiinsa liittyviä tietoja, jotka yhdessä kuvaavat halutun käsitteen ominaisuuksia. Koosteydinkomponentin sisältö muodostetaan toimialariippumattomista perus- ja assosiaatioydinkomponenteista (ks. Kuva 2). Koosteydinkomponentti vastaa objektiluokkaa, joka sisältää attribuutteja ja viitteitä muihin objekti-

luokkiin. Jokaisella koosteydinkomponentilla on yksilöivä objektiluokkatermi, jolla tietokomponenttiin viitataan. [UNC09]



Kuva 2: Koosteydinkomponentin rakenne

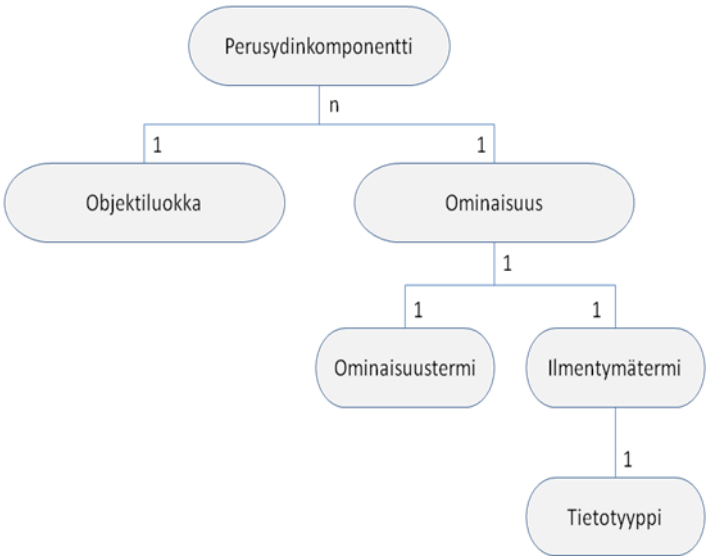
Esimerkiksi sosiaalihuollossa on määritelty koosteydinkomponentteja, joilla voidaan kuvata muun muassa ammattihenkilöitä, yhteystietoja ja organisaatioita. Koosteydinkomponentit sisältävät perusydinkomponentteja ja assosiaatioydinkomponentteja, joilla kuvataan koosteydinkomponenttien välisiä suhteita (Kuva 3).



Kuva 3: Sosiaalihuollossa määriteltyjä tietokomponentteja

Perusydinkomponentit (BCC) ovat koosteydinkomponenteille määriteltyjä ominaisuuksia, jotka esittävät yksittäisiä tietosisältöjä. Perusydinkomponentti muodostuu perusydinkomponentin ominaisuudesta (BCC Property) ja koosteydinkomponenttiin viittaavasta objektitermistä (Object Class Term) (Kuva 4). Perusydinkomponentin ominaisuus

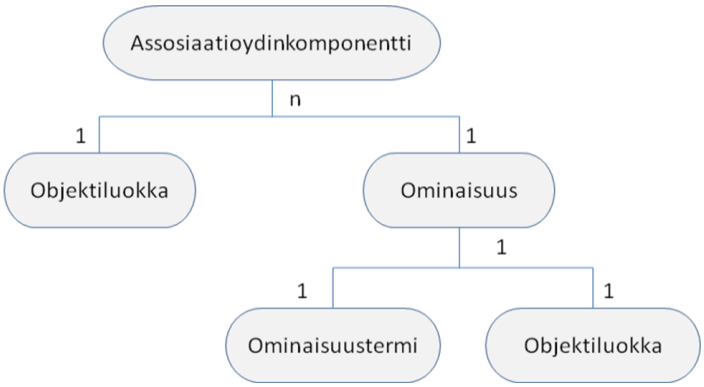
muodostuu ominaisuustermistä (Property Term) ja ilmentymätermistä (Representation Term). Perusydinkomponenttien ominaisuudet ovat yksittäistä tietosisältöä esittäviä uudelleenkäytettäviä ominaisuuksia, joiden arvo määritellään ydintietotyypillä. Samaa ominaisuutta voidaan käyttää useassa eri perusydinkomponentissa. [UNC09]



Kuva 4: Perusydinkomponentin rakenne

Esimerkiksi ammattihenkilöillä on ominaisuuksia, kuten henkilötunnus ja virkanimike. Nämä ominaisuudet määritellään `Ammattihenkilo`-koosteydinkomponentille perusydinkomponentteina. Henkilötunnus on yksi esimerkki uudelleenkäytettävästä ominaisuudesta, joka on käytössä myös `Yksityishenkilo`-koosteydinkomponentissa. (Kuva 3)

Assosiaatioydinkomponentilla (ASCC) kuvataan kahden koosteydinkomponentin välistä suhdetta. Assosiaatioydinkomponentti muodostuu viittauksesta objektiluokkaan ja uudelleenkäytettävästä ominaisuudesta (ASCC Property), joka määrittelee viittauksen muuhun koosteydinkomponenttiin (Kuva 5). Esimerkiksi `Ammattihenkilo`-tietokomponentilla on kaksi assosiaatioydinkomponenttia (Kuva 3), joilla viitataan koosteydinkomponentteihin `Organisaatio` ja `Yhteystiedot`.



Kuva 5: Assosiaatioydinkomponentin rakenne

Ominaisuustermeillä voidaan tarkentaa viitattavaa koosteydinkomponenttia. Esimerkiksi assosiaatioydinkomponentissa `Asiakas_ Yksityishenkilo`, ominaisuustermi `Asiakas_` tarkentaa `Yksitysihenkilo`-koosteydinkomponentin semanttista merkitystä. Assosiaatioydinkomponenttien assosiaatio-ominaisuutta voidaan uudelleenkäyttää samalla tavalla kuin perusydinkomponentin ominaisuuksia. Esimerkiksi yksityishenkilöä kuvaavalla koosteydinkomponentilla on vastaava assosiaatio yhteystietoihin.

2.5.1.2 Tietotyypit

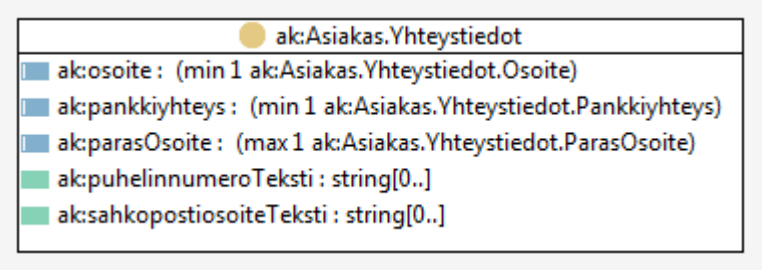
Tietotyypit edustavat perusydinkomponenttien ilmentymien sallittuja arvoalueita. Perusydinkomponenttien tietotyypit (Core Data Type) muodostuvat ISO 11179 -standardin [ISO10] mukaisesti sallituista arvoalueista (Value Domain). Kaikilla tietotyypeillä voi olla useita lisäattribuutteja, joilla voidaan määritellä tietotyyppien ilmentymille lisätietoja, kuten valuutta. [UNC09]

2.5.1.3 Toimialakohtaiset tietokomponentit

Toimialakohtaiset koostekomponentit (ABIE) johdetaan koosteydinkomponenteista (ACC) tarkentamalla koosteydinkomponenteissa määriteltyjä ominaisuuksia. Toimialakohtainen konteksti määritellään lisäämällä objektiluokkiin määretermit (Qualifiers), jotka tarkentavat tietokomponenttien käyttötarkoitusta. Toimialakohtaiselle koostekomponentille ja sen ominaisuuksille tulee kuvata toimialakohtaiset määritelmät ja esimerkit. [UNC09]

Toimialakohtaisessa koostekomponentissa ei tarvitse käyttää kaikkia koosteydinkomponenttien ominaisuuksia. Lisäksi koosteydinkomponentin ominaisuuksien kardinaliteetteja voi muuttaa. Määretermeillä voidaan tarkentaa koosteydinkomponenttien omi-

naisuuksien semanttisia merkityksiä. Toimialakohtaisen koosteydinkomponentin kenttiä voidaan myös tarkentaa määrittelemällä kentille toimialakohtaisia tietotyyppejä [UNC09]. Kuvassa 6 on esimerkki Tikesos-hankkeessa määritellystä Yhteystiedot-koosteydinkomponentista, jota on tarkennettu toimeentulotuen asiakkaan yhteystietoja kuvattaessa.



Kuva 6: Esimerkki toimialakohtaisesti tarkennetusta tietokomponentista

Toimialakohtainen assosiaatiokomponentti (ASBIE) kuvaa koostekomponentin (ABIE) käyttöä toisessa koostekomponentissa. Assosiaatiokomponentti sisältää vastaavan assosiaatioydinkomponentin tavoin ominaisuuden ja viittauksen toimialakohtaiseen koosteydinkomponenttiin. Assosiaatiokomponenttien ominaisuudet (ASBIE Property) ovat uudelleenkäytettäviä ominaisuuksia, joita voidaan käyttää useassa assosiaatiokomponentissa. Toimialakohtainen assosiaatiokomponentti määrittelee kardinaliteetit koostekomponentille, johon se perustuu. Määretermeillä voidaan tarkentaa viitattavan koosteydinkomponentin semantiikkaa. [UNC09]

Toimialakohtaiset perustietokomponentit (BBIE) johdetaan perusydinkomponenteista (BCC). Yhdestä perusydinkomponentista voidaan muodostaa monta erillistä perustietokomponenttia, joista jokaisella on oma määritelmä ja semanttinen merkitys. Perustietokomponentti muodostuu uudelleenkäytettävästä ominaisuudesta (BBIE Property) ja viittauksesta perustietokomponentin omistamaan koosteydinkomponenttiin. Määretermillä voidaan tarkentaa perusydinkomponentin semantiikkaa. [UNC09]

2.5.2 CCTS ja ISO/IEC 11179

ISO/IEC 11179 Metadata Registries -standardi [ISO10] (myöhemmin ISO 11179) määrittelee metatiedon rekisteröinti- ja kuvailumenetelmän, joka yhtenäistää metatiedon määrittelyn ja yksilöinnin metatietorekisterien näkökulmasta. Kuvailumenetelmä soveltuu tiedon ilmentymien, käsitteiden, tarkoitteiden ja niiden välisten suhteiden mallinta-

miseen. Standardin noudattaminen tukee tiedonhallintaa, tietomääritysten kehittämistä, tiedonvälitystä organisaatioiden välillä ja tiedon uudelleenkäyttöä.

ISO 11179 -standardin mukaan metatieto on tietoa, joka määrittelee ja kuvaa muuta tietoa [ISO10]. Metatietoa voidaan hallita ja organisoida siihen tarkoitukseen suunnitelluissa järjestelmissä. Metatietorekisteri on metatiedon hallintajärjestelmä, jolla metatietoa voidaan kuvata ja rekisteröidä yksilöivästi.

Metatiedon käyttötarkoituksen ja historian tunteminen on olennaista metatietojen hallinnassa. Metatiedon rekisteröinti mahdollistaa tiedon yksilöinnin ja hallinnan. Tiedon merkityksen ymmärtäminen on lähtökohta tiedonvälitykseen, tietojen mallintamiseen, standardointiin ja uudelleenkäyttöön.

Metatietorekisterin metatietomalli määrittelee metatietorekisterissä käytettävien tietosisältöjen tarkan rakenteen ja suhteet toisiinsa. Metatietomalli määrittelee kieliopin, jonka avulla voidaan mallintaa eri toimijoiden ja organisaatioiden tietosisältöjä. Metatietomalli ei rajoita metatietorekisterin teknistä toteutusta, joten tietosisältöjä voidaan mallintaa useilla olemassa olevilla tekniikoilla, kuten ontologioilla tai tietokannoilla.

CCTS pohjautuu ISO/IEC 11179 -standardiin, ja sitä voidaan pitää yhtenä standardinmukaisen metatietomallin toteutuksista. Koosteydinkomponentit vastaavat ISO 11179 -standardin mukaisia objektiluokkia, perusydinkomponentit ja assosiaatioydinkomponentit objektiluokan ominaisuuksia ja ilmentymiä.

ISO 11179 -standardi luo yhtenäisen pohjan käsitteiden, termien, määritelmien ja tietoyksiköiden arvoalueiden muodostamiselle. Yhtenäisen metatietomallin käyttö (kuten CCTS) on edellytys eri toimijoiden väliselle automatisoidulle viestinvälitykselle.

2.6 Sosiaalihuollon asiakastietomalli

Sosiaalihuollon asiakastietomalli on kehitetty CCTS-menetelmän mukaisesti, yhtenäistämällä eri palvelutehtävissä käytettäviä tietosisältöjä. Asiakastietomalli on tuotettu yhteistyössä sosiaalihuollon sisällön- ja teknisten asiantuntijoiden kanssa. Tietosisältöjen yhtenäistäminen on toteutettu muodostamalla tietokomponentteja asiakirjojen tietosisältöjen pohjalta. Kaikkien asiakirjojen yhteisistä tietosisällöistä on muodostettu yhtenäisiä tietokomponentteja. Yhtenäinen tietomalli mahdollistaa viestinvälityksen ja asiakastie-

tojen uudelleenkäytön asiakasasiakirjoissa ja asiakastietojärjestelmissä. [SAA10, SYS09]

Sosiaalihuollon asiakastietomalli koostuu tietokomponenteista ja palvelutehtäväkohtaisista asiakirjoista (Kuva 7).

Kuva 7: Esimerkki sosiaalihuollon asiakastietomallin mukaisista tietokomponenteista

Tikesos-hankkeessa on tunnistettu 22 sosiaalihuollon palvelutehtävää ja 62 palvelua. Sosiaalihuollon asiakirjojen tietosisällöt on yhtenäistetty tietokomponenteiksi ja asiakirjakohtaisesti tarkennetuiksi tietokomponenteiksi. Asiakastietomallissa on yhteensä 148 ydintietokomponenttia ja niitä käytetään tarkennettuina noin 217 asiakasasiakirjassa.

Sosiaalihuollon asiakirjat muodostuvat asiakirjakohtaisesti tarkennetuista tietokomponenteista sekä asiakirjakohtaisista perustietokentistä. Asiakirjat on luokiteltu Tikesos-hankkeessa määritellyn asiakirjatyyppiluokituksen mukaisesti [HLT+11]. Yleisiä asiakirjatyyppejä on yhteensä 15:

- Arvio
- Hakemus
- Ilmoitus
- Kertomus
- Laskelma
- Lausunto
- Lähete
- Pyyntö
- Päätös
- Selostus
- Selvitys
- Sitoumus
- Sopimus
- Suostumus
- Suunnitelma

Asiakirjoille määritellään palvelutehtäväkohtaisesti tarkennetut tyypit. Esimerkiksi toimeentulotuessa yleinen asiakirjatyyppi on Hakemus ja tarkennettu asiakirjatyyppi Toimeentulotukihakemus. Asiakirjatyyppiluokitusta hyödynnetään asiakirjojen tunnistamisessa ja nimeämisessä. Yleisten asiakirjatyyppien pohjalta on luotu ydintietokomponentit, joita voidaan hyödyntää asiakirjamallinnuksessa. Asiakirjamallinnuksessa on havaittu, että tarkennetut asiakirjatyypit eivät usein noudata yleisen asiakirjatyypin rakennetta. [HLT+11, HHM+09]

3 SEMANTTINEN MALLINTAMINEN

Semanttisella mallintamisella tarkoitetaan ihmisten *tietämyksen* kuvaamista käsitemalleina, jotka esittävät käsitteitä ja niiden välisiä suhteita. Käsitemalleja kutsutaan usein myös ontologioiksi. Ontologia on filosofian osa-alue, jossa tutkitaan olemassaolon käsitteitä, joiden avulla pyritään hahmottamaan kaikkea olemassa olevaa. Tietojenkäsittelytieteessä ontologialla tarkoitetaan yleensä formaalisti määriteltyä tietomallia käsitteiden välisistä suhteista [Ber02].

Käsitteitä voidaan mallintaa useaan eri tarkoitukseen käyttötarpeiden mukaan:

1. Kommunikointi - Informaationvälitys ihmisten välillä
2. Tiedonkäsittely - Tietämyksen määrittely ja koneellinen päättely
3. Yhteentoimivuus - Tietojärjestelmien välinen viestinvälitys

Informaationvälitykseen tarkoitetut abstraktit käsitemallit mahdollistavat käsitteiden vapaamman ilmaisun, mutta jättävät niiden tulkinnan ihmisen varaan. Abstrakti käsitemalli voi olla kirjallinen tai kuvallinen esitys. Esimerkiksi lakeja voidaan pitää abstrakteina käsitemalleina, joiden tulkitseminen jätetään ihmisten tehtäväksi.

Tietämyksen määrittely formaaleiksi ontologioiksi mahdollistaa koneellisen päättelyn logiikan avulla. Formaalit tietomallit edellyttävät käsitteiden tarkempaa määrittelyä kuin informaationvälitykseen tarkoitetut käsitemallit. Formaaleilla kielillä mallintaminen ei kuitenkaan takaa koneellisesti pääteltävissä olevaa lopputulosta, jos käsitteiden välisiä suhteita ei ole mallinnettu riittävän tarkalla tasolla [AlH08]. Semanttiset teknologiat perustuvat käsitteiden formaaliin mallintamiseen ja käsitteistä johdettaviin päätelmiin ja oletuksiin.

Tietosisällöistä voidaan tehdä loogisia päätelmiä tiettyjen oletusten ja sääntöjen perusteella. Oletukset voidaan jakaa kahteen pääryhmään, suljetun ja avoimen maailman oletuksiin. Suljetun maailman oletuksella tarkoitetaan sitä, että saatavilla oleva tieto oletetaan muuttumattomaksi ja täydelliseksi. Avoimen maailman oletuksen mukaan tieto voi olla epätäydellistä tai jopa virheellistä. [SOW99, Rei78]

Ontologiat perustuvat yleensä avoimen maailman oletukselle, koska kaikkea tietoa on mahdotonta mallintaa yksiselitteisesti. Ontologioiden tietosisältö koostuu usein erillisistä ontologioista, jotka voivat sisältää puutteellisia, päällekkäisiä ja ristiriitaisia tietoja.

Avoimen maailman oletuksen mukaisesti tämä ei ole kuitenkaan ongelma, ja tietosisällöstä voidaan johtaa uutta tietoa määriteltyjen sääntöjen mukaisesti. [SOW99]

Tässä tutkimuksessa semanttisia teknologioita sovelletaan sosiaalihuollon asiakirjojen mallintamiseen. Asiakirjamallinnus ja asiakirjasisältöjen validointi perustuvat suljetun maailman oletukseen. Voidaan siis olettaa, että asiakirjan sisältö on saatavilla kokonaisuudessaan tietyllä ajan hetkellä ja että poikkeamat tietosisällössä oletetaan virheiksi. Suljetun maailman oletus on ristiriidassa semanttisen webin periaatteiden kanssa, minkä vuoksi asiakirjamallinnukseen ei voida suoraan soveltaa kaikkia semanttisen webin teknologioita.

Tässä luvussa esitetään semanttisten kielten periaatteet ja kuvataan miten RDF-kielioppeja voidaan laajentaa. Suljetun oletuksen mukaisen validoinnin toteuttamiseksi semanttisten teknologioiden avulla esitellään RDF-metatietomalli ja säännöstö, joka mahdollistaa asiakirjasisältöjen validoinnin. Metatietomalli kuvaa CCTS-standardin mukaisia tietorakenteita RDF-muodossa. CCTS-metatietomallin toteutus on esitetty luvussa 3.2.

3.1 **Semanttiset teknologiat**

Semanttiset teknologiat koostuvat kielistä ja tekniikoista, jotka toimivat perustana semanttiselle webille. Semanttinen web on perinteisen webin laajennus, jossa tietosisällöille määritelty semantiikka mahdollistaa automaattisen tiedonkäsittelyn ja päättelyn. Suurinta osaa nykyisen webin sisällöstä ei ole tarkoitettu tiedon automaattiseen käsittelyyn. Tietosisältöjen merkityksen lisääminen web-sivustoille mahdollistaa tiedon uudelleenkäytön eri käyttötarkoituksissa. Kun web-sivustojen tietosisältöjen semantiikka on selkeästi määritelty, voidaan sivustoilla kuvattua tietoa yhdistämällä ja päättelemällä luoda kokonaan uutta tietoa. [Ber02]

Nykyinen web on muodostunut miljardien ihmisten osallistumisesta informaation tuottamiseen ja sen julkaisemiseen webissä. Semanttinen web keskittyy olemassa olevan tiedon merkitysten määrittelemiseen tiedon esitysmuotojen määrittelyn sijaan. Semanttisen sisällön tuottaminen ja käsittely toimii kuitenkin samojen pääperiaatteiden mukaisesti, jotka on tunnistettavissa perinteisestä webistä [AlH08]:

1. Kuka tahansa voi sanoa mitä tahansa mistä tahansa

   Eri toimijoiden välillä voi olla suuria erimielisyyksiä tärkeistäkin asioista. Samasta asiasta voi olla myös useita virallisia määritelmiä. Ristiriitaiset tiedot voivat olla vanhentuneita, vahingossa tuotettuja tai tarkoituksenmukaisesti väärin määriteltyjä.

2. Erinimiset käsitteet voivat tarkoittaa samaa asiaa (Nonunique Naming Assumption)

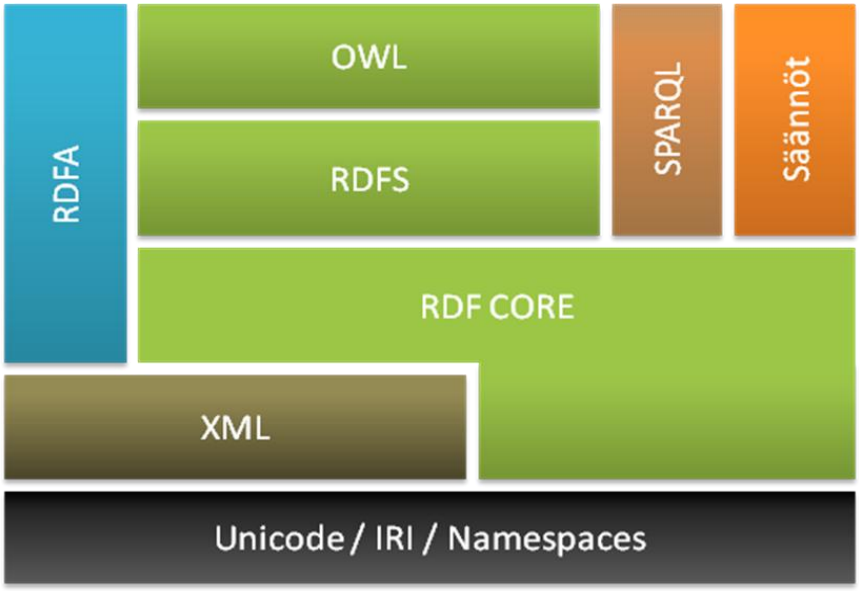
   Tietoja määritellään hajautetusti usealla eri taholla, joten samojen asioiden määrittely johtaa eri tavoin ilmaistuihin tietoihin, joilla kuitenkin tarkoitetaan samoja käsitteitä. Samaa tarkoittavia käsitteitä voidaan nimetä monella eri tavalla. Erinimisten käsitteiden tarkoitusta ei voida siis erottaa toisistaan, ellei eroavaisuutta ole eksplisiittisesti määritelty.

3. Puuttuva tieto voi olla määritelty jossain muualla (Open World Assumption)

   Hajautetun tietomallin seurauksena ei voida tehdä oletuksia puutteellisesta tiedosta. Yhdellä hetkellä haettu tieto voi olla kohta päivittynyt tai jonkin verkkopalvelun käyttö on estynyt, kun tietoa on haettu.

Semanttisia teknologioita kuvataan usein semanttisen webin teknologiapinoilla. Teknologiapinoilla voidaan havainnollistaa hyvin semanttisten teknologioiden riippuvuuksia toisistaan. Semanttisen webin rakentuminen pienemmistä osista mahdollistaa teknologioiden nopeamman kehityksen, koska pienemmistä osa-alueista päästään helpommin yhteisymmärrykseen. Teknologiat on yleensä pinottu siten, että jokaisella tasolla on mahdollista tehdä toteutuksia, jotka pystyvät käyttämään osittain ylemmillä tasoilla määriteltyjä standardeja ja kokonaan alemmalla tasolla määriteltyjä standardeja [AnH04].

Semanttisia teknologiapinoja on kymmenen vuoden aikana muodostettu useita [HPP+05, KIF05]. Vaikka eri teknologiapinoilla on korostettu erilaisia näkemyksiä, perusperiaatteet ovat olleet kuitenkin samoja. Kuvassa 8 on esitetty standardisoituja semanttisten webin teknologioita, jotka ovat käyttövalmiita ja tämän tutkielman kannalta olennaisia.



Kuva 8: Semanttisen webin teknologiat

Semanttisten teknologioiden perustana on RDF, jonka toiminta perustuu Unicode-merkistöön ja IRI (Internationalized Resource Identifier) -tunnisteisiin. Tunnisteiden avulla yksilöidään kaikki RDF-resurssit. RDF käyttää XML:n tavoin nimiavaruuksia (Namespaces), jotka määritellään IRI-tunnisteilla. Nimiavaruudet yksilöivät eri tietomalleissa määritellyt samannimiset käsitteet, joka ehkäisee ristiriitoja useita RDF-tietomalleja yhdistäessä. RDF-graafeja voidaan myös kuvata ja käsitellä XML-tiedostoina RDF/XML-muodossa (Luku 3.1.1). RDFa:n (Luku 3.1.4) toimintaperiaate perustuu XML-asiakirjojen annotointiin ja mahdollistaa tietosisältöjen semanttisen merkkauksen. RDFS (Luku 3.1.2) ja OWL (Luku 3.1.3) ovat RDF-kielen avulla määriteltyjä kielioppeja, jotka mahdollistavat luokittelun ja semantiikan määrittelyn RDF-kielen avulla. SPARQL (Luku 3.1.5) on RDF-kielelle määritelty kyselykieli, jonka avulla on mahdollista muodostaa kyselyitä RDF-tietomalleihin. RDFS-tietomallien ja OWL-ontologioiden ilmaisuvoima ja päättelykyky perustuu formaaleihin rakenteisiin ja sääntöihin, joiden perusteella eksaktisti määritellystä tiedosta voidaan johtaa uutta tietoa. Sääntöjen mallintamiseen on määritelty W3C:n suositus RIF (*Rule Interchange Format*) [RIF10]. RIF-suositusten tarkoituksena on yhtenäistää eri sääntökoneiden käyttämien kielien syntaksi ja kuvata RDF/OWL -ontologioiden päättelysäännöt standardilla tavalla. Tässä tutkielmassa sääntökieleksi on kuitenkin valittu avoimen lähdekoodin JenaRules (Luku 4.2.1), koska avoimen lähdekoodin ratkaisuja RIF-sääntöjen käyttöön ei ollut vielä saatavilla.

3.1.1 RDF

Semanttisten teknologioiden perusta on W3C:n standardisoima RDF (Resource Description Framework), joka on laajennettavissa oleva abstrakti metatietomalli tiedon mallintamiseen. Suositus määrittelee yksiselitteiset periaatteet resurssien ja niiden välisten suhteiden esittämiseen. [RDF01]

Keskeisenä ajatuksena RDF-mallissa on, että kaikki tieto voidaan esittää yksinkertaisina ”subjekti-predikaatti-objekti”-lausekkeina. Lausekkeet muodostuvat resursseista ja literaaleista. RDF muodostaa suunnatun graafin, jossa kaikki resurssit on yksilöity IRI (Internationalized Resource Identifier) -tunnisteilla. RDF-tietomallin peruselementtejä ovat:

- **Lauseke** (Statement)
  Lausekkeella tarkoitetaan kokonaista ”subjekti-predikaatti-objekti” -lauseketta, jossa jokin resurssi viittaa predikaatilla toiseen resurssiin tai literaaliin.
- **Predikaatti** (Property)
  Predikaatilla tarkoitetaan subjektin ominaisuutta tai suhdetta kahden resurssin välillä.
- **Resurssi** (Resource)
  Resurssilla tarkoitetaan tietoa, joka voidaan yksilöidä URI-tunnisteella. Resurssi voi olla web-sivusto, kirja tai mikä tahansa koodi, esimerkiksi paikkatunniste.
- **Literaali** (Literal)
  Literaalilla tarkoitetaan teksti- tai numeromuotoista arvoa, jolle voidaan määritellä tietotyyppi.

Yksinkertaisin syntaksi RDF-lausekkeille on N-Triples-muoto. Alla olevassa esimerkissä on kuvattu henkilö, jonka etunimi on ”Erkki”.

```
<http://sosmeta.fi/sos#Henkilo>
<http://sosmeta.fi/sos#etunimi> "Erkki" .
```

Semanttisessa webissä käytetään yleisesti IRI-tunnisteina oman organisaation URL-osoitetta tai kuvitteellisia osoitteita. URL-osoitteiden käyttö RDF-tietomallissa ei edellytä, että kyseinen resurssi olisi saatavilla nimetystä osoitteesta. Piste lausekkeen perässä erottaa lausekkeet toisistaan. Tässä tutkielmassa RDF-esimerkkien esittämiseen käy-

tetään yksinkertaisempaa N3-esitysmuotoa. N3 lisää esitysmuotoon etuliitteen (prefix), joka piilottaa IRI-tunnisteet.

N3-muoto:

```
@prefix sos: <http://sosmeta.fi/sos>.
sos:Henkilo sos:etunimi "Erkki" .
sos:Henkilo sos:sukunimi "Esimerkki".
sos:Henkilo sos:asuu sos:Osoite .
sos:Osoite sos:katu "Esimerkintie 1".
```

Edellisessä esimerkissä kuvatun henkilön tiedot voidaan esittää monella eri tavalla, koska RDF on abstrakti metakieli, jolle on määritelty useita eri esitystapoja ja tiedostomuotoja. Esimerkki voidaan esittää myös RDF/XML-muodossa:

```
<?xml version="1.0" encoding="UTF-16"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:sos="http://sosmeta.fi/sos#">
    <rdf:Description rdf:about="sos:Henkilo">
        <sos:asuu rdf:resource="sos:Osoite"/>
        <sos:etunimi>Esimerkki</sos:etunimi>
        <sos:sukunimi>Erkki</sos:sukunimi>
    </rdf:Description>
    <rdf:Description rdf:about=”sos:Osoite”>
        <sos:katu>Esimerkintie 1</sos:katu>
    </rdf:Description>
</rdf:RDF>
```

RDF-standardi määrittelee rakenteita, joilla voidaan mallintaa ja organisoida tietoa [RDF01]. RDF-kielen rakenteita voidaan laajentaa käyttämällä RDF-kieltä. Esimerkiksi RDFS ja OWL ovat RDF-kielen semanttisia laajennuksia, jotka on määritelty RDF-kielen avulla.

3.1.2 RDFS

RDFS (Resoruce Description Framework Schema) lisää RDF-määrityksiin yksinkertaisen kieliopin luokille ja luokkien välisille suhteille. RDFS-kielen tärkeimmät käsitteet ovat `rdfs:Class` ja `rdfs:subClassOf`. Näiden laajennusten avulla aiempiin esimerkkeihin voidaan lisätä semantiikkaa ja tehdä sen perusteella loogista päättelyä.

RDFS-rakenteiden avulla voidaan määritellä luokka ja sille ilmentymä:

```
sos:Yksityishenkilo rdf:type rdfs:Class .
esim:Erkki rdf:type sos:Yksityishenkilo .
```

Lausekkeet määrittelevät, että `sos:Yksityishenkilo` on luokka, ja `esim:Erkki` on luokan `sos:Yksityishenkilo` ilmentymä. Luokkien välille voidaan määritellä myös riippuvuussuhteita `rdfs:subClassOf`-rakenteen avulla:

```
sos:Ihminen rdf:type rdfs:Class .
sos:Yksityishenkilo rdfs:subClassOf sos:Ihminen .
```

Tämä määrittelee, että `sos:Yksityishenkilo` on luokan `sos:Ihminen` aliluokka. Kun määritellään `sos:Yksityishenkilo` luokalle ilmentymä, voidaan koneellisesti päätellä, että ilmentymä kuuluu myös luokkaan `sos:Ihminen`:

```
esim:Erkki rdf:type sos:Yksityishenkilo
   ➔  esim:Erkki rdf:type sos:Ihminen
```

RDFS mahdollistaa myös predikaattien määrittelemisen luokan ominaisuutena `rdfs:domain`-viittauksen avulla. Predikaateille voidaan määritellä myös arvoalue `rdfs:range`-viittauksella. Esimerkiksi yksityishenkilölle voidaan määritellä ominaisuus etunimi määrittelemällä seuraavat lausekkeet.

```
sos:etuNimi rdf:type rdf:Property
sos:etuNimi rdfs:domain sos:Yksityishenkilo
sos:etuNimi rdfs:range xsd:String
```

RDFS-kielen ilmaisuvoima on kuitenkin rajallinen, koska kieli on tarkoitettu lähinnä uusien luokkien määrittelyyn, tietojen luokitteluun sekä ryhmittelyyn. RDFS-kielen avulla ei voida määritellä luokkien ja niiden välisten suhteiden semantiikkaa.

3.1.3 OWL

OWL (Web Ontology Language) määrittelee joukon RDF-rakenteita, joilla voidaan ilmaista monipuolisia RDF-resurssien välisiä semanttisia suhteita. OWL-suositus määrittelee rakenteiden semanttisen merkityksen ja joukon sääntöjä, joiden avulla voidaan esimerkiksi [AnH04]:

- Ryhmitellä luokkia ja ilmentymiä aksioomien avulla
- Määritellä ekvivalenssiluokkia
- Määritellä kardinaliteetteja

- Johtaa uutta tietoa määriteltyjen luokkien ja suhteiden avulla
- Havaita epäjohdonmukaisuuksia

OWL-ontologioiden ilmaisuvoima perustuu formaaliin logiikkaan, jolla on määritelty päättelysääntöjä kielen rakenteille. OWL-kielessä on kaksi valmiiksi määriteltyä luokkaa, `owl:Thing` ja `owl:Nothing`. Ontologioissa resurssit määritellään alakäsitteiksi `owl:Thing`-käsitteelle, joka on kaikkien resurssien yläkäsite.

Uudet käsitteet määritellään ontologiaan `owl:Class`-tyyppisinä luokkina, objektisuhteet `owl:ObjectProperty`- ja luokkien muuttujat `owl:DatatypeProperty`-tyyppisinä suhteina. OWL 2 -suositus määrittelee useita rakenteita, joiden avulla voidaan määritellä käsitteiden ja niiden välisten suhteiden semantiikka formaalilla tavalla. Tässä kappaleessa esitetään esimerkkien avulla tämän tutkimuksen kannalta oleelliset OWL-kielen rakenteet.

Esimerkiksi yksityishenkilö, henkilötunnus ja ihminen voidaan määritellä ontologiassa:

```
sos:Yksityishenkilo rdf:type owl:Class
sos:henkiloTunnus rdf:type owl:DatatypeProperty
sos:Ihminen rdf:type owl:Class
```

OWL-kielellä voidaan määritellä luokille myös rajoituksia `owl:Restriction`-luokan avulla. Rajoitusten avulla voidaan määritellä sääntöjä, kuten ”Jokaisella yksityishenkilöllä on aina vain yksi henkilötunnus.”:

```
sos:Yksityishenkilo rdfs:subClassOf :A1
:A1 rdf:type owl:Restriction
:A1 owl:onProperty sos:henkiloTunnus
:A1 owl:cardinality "1"
```

Rajoitukset muodostavat nimettömiä solmuja, joissa on viittaus rajoittavaan predikaattiin ja rajoitettuun arvoon. Nimettömät solmut yksilöidään automaattisesti generoitavilla tunnuksilla. Rajoituksten avulla voidaan päätellä uutta tietoa luokkien ilmentymistä. Jos ilmentymä täyttää jossain toisessa luokassa määriteltyjen rajoituksten asettamat ehdot, voidaan päätellä, että se on myös kyseisen luokan ilmentymä. Kyseinen päätelmä tehdään avoimen oletuksen mukaisesti aina, jos luokkien ei ole erikseen määritelty eroavan toisistaan. Esimerkiksi jos `sos:Ihminen` luokan ilmentymälle olisi määritelty henki-

lötunnus, voitaisiin ilmentymästä päätellä, että se on myös luokan `sos:Yksityishenkilo` ilmentymä.

3.1.4 RDFa

RDFa (Resource Description Framework Attributes) on W3C:n suositus, joka mahdollistaa XHTML-kielellä esitettyjen asiakirjojen tietosisältöjen annotoinnin [RDA08]. RDFa määrittelee joukon attribuutteja, joilla voidaan lisätä semantiikkaa XHTML-dokumenttien elementeille. RDFa-merkkauksella voidaan esimerkiksi annotoida XHTML-sivu, jossa esitetään tietoja aiemmissa esimerkeissä tutuksi tulleesta yksityishenkilöstä:

```
<div about="esim:Erkki" typeof="sos:Yksityishenkilo">
  <span property="sos:etunimi">Erkki</span>
  <span property="sos:sukunimi">Esimerkki</span>
  <div rel="sos:asuu">
    <span property="sos:katu">Esimerkintie 1</span>
  </div>
 <p>Muuta vähäpätöisempää tietoa Erkistä</p>
</div>
```

Sosiaalihuollon asiakirjoissa käytettävän XHTML-merkkauksen kannalta oleelliset RDFa-attribuutit ovat:

@typeof-attribuutilla voidaan määritellä subjektin luokka.

@property-attribuutilla voidaan määritellä subjektin ja literaalin välinen suhde.

@rel-attribuutilla voidaan määritellä subjektin ja objektin välinen suhde.

@rev-attribuutilla voidaan määritellä objektin ja subjektin käänteinen suhde.

@content-attribuutilla voidaan määritellä koneellisesti luettavan literaalin arvo.

@about-attribuutilla voidaan määritellä subjektin nimi IRI-tunnisteen avulla.

RDFa määrittelee myös joukon muita attribuutteja, joilla voidaan määritellä esimerkiksi web-sivujen välisten linkkien tarkempaa semantiikkaa [RDA08].

Edellisestä XHTML+RDFa-esimerkistä voidaan jäsentää RDF-graafi esimerkiksi XHTML-rakenteiden validoinnin yhteydessä. W3C:n RDF Core määrittelee RDF:n

jäsentämisen käsittelysäännöt [RDC10]. Esimerkin mukaisen XHTML+RDFa-rakenteen jäsentäminen muodostaa seuraavan RDF-graafin:

```
esim:Erkki rdf:type sos:Yksityishenkilo
esim:Erkki sos:etunimi "Erkki" .
esim:Erkki sos:sukunimi "Esimerkki".
esim:Erkki sos:asuu :N1 .
:N1 sos:katu "Esimerkintie 1".
```

Jäsentämisen tuloksena syntynyt graafi sisältää kaikki XHTML+RDFa-asiakirjassa annotoidut tiedot. Graafissa on yksi nimetön solmu `:N1`, jolle ei esimerkissä määritelty nimeä tai luokkaa. Kaikkia objekteja ei ole välttämätöntä nimetä. RDF-jäsennin muodostaa nimettömistä objekteista resursseja, joille generoidaan yksilöllinen tunniste.

XHTML+RDFa-rakenne voidaan validoida, mutta RDF sisällön validointia ei voida toteuttaa perinteisellä XML-skeemalla. Tutkimuksessa on kuitenkin kehitetty menetelmä XHTML+RDFa-asiakirjojen RDF-sisällön validointiin (Luku 4).

3.1.5 SPARQL

SPARQL on W3C:n standardoima kyselykieli RDF-tietomallille. SPARQL-kyselyt voidaan muodostaa useista lausekkeista ja niitä yhdistävistä operaatioista. Kyselyillä palautetaan arvoja muuttujille, jotka määritellään kyselyn alussa kysymysmerkeillä. Kyselyn tuloksena muuttujan sisällöksi asetetaan RDF-tietomallista muuttujaa vastaava resurssi. [SRQ08]

SPARQL-kielessä on neljä erityyppistä kyselyä:

- SELECT-kysely palauttaa määritellyt muuttujat kyselyssä annettujen lausekkeiden perusteella. Esimerkiksi seuraava kysely palauttaa resurssin, jonka tyyppi (`rdf:type`) on `sos:Henkilo` ja `sos:etuNimi` on `"Erkki"`.

  ```
  SELECT ?esimerkki WHERE {
          ?esimerkki rdf:type sos:Henkilo .
          ?esimerkki sos:etuNimi "Erkki" }
  ```

- CONSTRUCT-kysely palauttaa CONSTRUCT-osassa määriteltyjen lausekkeiden mukaisen graafin, kysely-osassa määriteltyjen lausekkeiden löytyessä RDF-tietomallista. Esimerkiksi seuraava kysely muodostaa uuden lausekkeen löydet-

tyjen resurssien pohjalta ja asettaa löydetylle resurssille viittauksen `sos:sukuNimi` ja sille arvon `"Esimerkki"`:

```
CONSTRUCT {  ?esimerkki sos:sukuNimi "Esimerkki" }
WHERE     {  ?esimerkki rdf:type sos:Henkilo .
             ?esimerkki sos:etuNimi "Erkki" .    }
```

- ASK-kysely palauttaa totuusarvon määriteltyjen lausekkeiden löytyessä. Esimerkiksi seuraava kysely palauttaa totuusarvon tosi, jos RDF-tietomallista löytyy resurssi, jolle on määritelty `sos:etunimi "Erkki"` ja `sos:sukuNimi "Esimerkki"`:

```
ASK  { ?esimerkki sos:etuNimi  "Erkki" .
       ?esimerkki sos:sukuNimi ”Esimerkki” }
```

- DESCRIBE-kysely palauttaa kaikki jollekkin resurssille määritellyt lausekkeet. Esimerkiksi seuraava kysely palauttaa kaikki lausekkeet, jotka liittyvät resurssiin, jolle on määritelty `sos:etuNimi "Erkki"`:

```
DESCRIBE ?erkki WHERE { ?erkki sos:etuNimi "Erkki" }
```

SPARQL-kyselyille voidaan asettaa myös erilaisia jälkiehtoja, joiden avulla tuloksia voidaan suodattaa tai järjestellä. SPARQL-kielen tarkempi kuvaus on määritelty suosituksessa [SQR08]. Kyselyiden vastaussanomille on useita eri muotoja, kuten W3C:n standardoimat JSON ja XML-muotoiset vastaussanomat. Esimerkiksi aiemmin esitetyn SELECT-kyselyn esimerkin vastaus XML-muotoisena voisi olla seuraava:

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="esimerkki"/>
  </head>
  <results>
    <result>
      <binding name="esimerkki">
       <literal
datatype="http://www.w3.org/2001/XMLSchema#string">Erkk
i</literal>
      </binding>
    </result>
  </results>
</sparql>
```

SPARQL-kyselyitä ja XML-muotoisia vastaussanomia voidaan hyödyntää XSLT-muunnostiedostoissa, esimerkiksi XML-skeemojen generointiin RDF-tietomallista. XSLT-muunnostiedostossa voidaan muodostaa tilanteen mukaan muuttuvia SPARQL-kyselyitä. XSLT/SPARQL-kyselyissä sovelletaan Burretan ym. [BLH08] kuvaamaa menetelmää. Seuraava XSLT/SPARQL-kysely hakee kaikki Vesa-Matti Loirin tähdittämät elokuvat julkisesta dbpedia-tietokannasta [DBP11] ja muodostaa niistä HTML-sivun.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:results="http://www.w3.org/2005/sparql-results#"
version="2.0">
  <xsl:output method="html" version="4.0" encoding="ISO-8859-1"
indent="yes"/>
  <xsl:variable name="sparqlEndpoint"
select="'http://dbpedia.org/sparql/?query='"/>
  <xsl:variable name="XMLMIMEType"
select="'&amp;Accept=application/sparql-results%2Bxml'"/>
  <xsl:variable name="prefix">
PREFIX rdfs:&lt;http://www.w3.org/2000/01/rdf-schema#&gt;
PREFIX xsd:&lt;http://www.w3.org/2001/XMLSchema#&gt;
PREFIX rdf:&lt;http://www.w3.org/1999/02/22-rdf-syntax-ns#&gt;
PREFIX db:&lt;http://dbpedia.org/ontology/&gt;
PREFIX purl:&lt;http://purl.org/dc/terms/&gt;
PREFIX res:&lt;http://dbpedia.org/resource/&gt;
</xsl:variable>
  <xsl:variable name="query">
    <xsl:value-of select="$prefix"/>
      SELECT ?label
      WHERE {
              ?film rdfs:label ?label .
              ?film db:starring res:Vesa-Matti_Loiri
      FILTER (LANG(?label) = 'fi') .
            }
  </xsl:variable>
  <xsl:variable name="endpointResponse"
select="doc(concat($sparqlEndpoint,encode-for-
uri($query),$XMLMIMEType))"/>
  <xsl:template match="text()"/>
  <xsl:template match="/">
    <html>
      <body>
    <h1>Veskun leffat</h1>
        <ul><xsl:apply-templates
select="$endpointResponse/results:sparql"/></ul>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="results:result">
    <li>
     <xsl:value-of select="results:binding[@name='label']"/>
    </li>
  </xsl:template>
</xsl:stylesheet>
```

3.2 CCTS-mallin ontologisointi

CCTS on teknisestä esitysmuodosta riippumaton tietosisältöjen mallinnusmenetelmä, jolle ei ole määritelty virallista RDF-esitysmuotoa. Eri käyttötarkoituksiin on kuitenkin mallinnettu ontologioita, joissa käytetään CCTS-menetelmässä määriteltyjä käsitteitä. Tässä luvussa esitetään tietokomponenttien ja asiakirjojen mallintamiseen ja validointiin uusi ontologiapohjainen menetelmä, joka soveltaa CCTS-spesifikaation mukaisia määrittelyjä.

CCTS-ontologisoinnin tavoitteiksi asetettiin tietokomponenttien hallinta ja asiakirjasisältöjen validoinnin mahdollistaminen RDF-tietorakenteista. Ontologiapohjaisessa asiakastietomallissa ja XHTML+RDFa-muotoisissa asiakirjoissa nähtiin ratkaisu pitkään vireillä olleeseen ajatukseen näyttömuotojen ja tietorakenteiden yhdistämisestä.

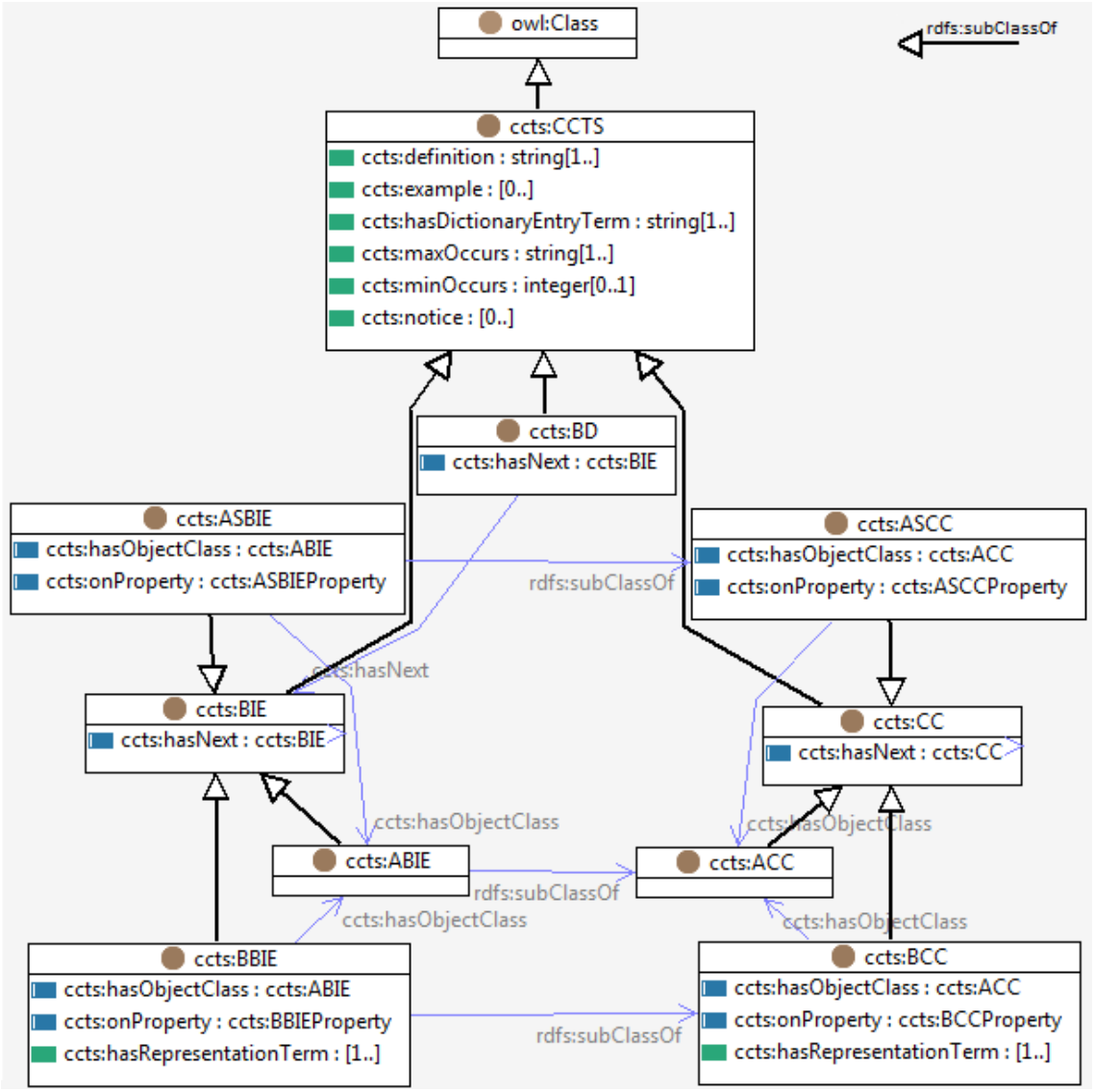
Ontologian mallintamiselle tulee olla selkeät lähtökohdat ja käyttötarkoitus, joiden avulla voidaan perustella käytettäviä rakenteita. Tässä tutkimuksessa ontologiaa lähdettiin kehittämään metatietorekisterin näkökulmasta. Metatietorekisterin tarkoituksena on hallita tietokomponenttikirjastoa ja asiakirjarakenteita sekä mahdollistaa XML-skeemojen automaattinen generointi. Ennen sosiaalihuollon asiakastietomallin ontologisointia kartoitettiin alustavasti CCTS-menetelmää soveltavia tietojärjestelmiä.

Tutkimuksen kannalta kiinnostavin tietojärjestelmä oli Hollannin oikeusministeriön kehittämä MDW (Metadata Workbench) [BiH10], joka on semanttisia teknologioita hyödyntävä metatietorekisteri. Sitä hyödynnetään oikeusministeriön hallinnonalalla käytettävien tietokomponenttien ja asiakirjojen mallintamiseen ja ylläpitoon. MDW mahdollistaa CCTS-menetelmän mukaisen tietokomponenttien muodostamisen OWL-ontologioista. Ontologioista muodostetuista tietokomponenteista voidaan edelleen muodostaa asiakirjoja ja generoida skeemoja. MDW-järjestelmän tietomalli laajentaa RDF-tietomallia CCTS-menetelmän mukaisilla metatiedoilla. RDF-kielen laajentaminen uusilla rakenteilla mahdollistaa tietokomponenttien mallintamisen luokkatasolla, jolloin asiakirjat voidaan mallintaa käyttämällä näitä luokkia. Tämä mahdollistaa myös sen, että RDF-kielen ilmentymätasolla voidaan käsitellä asiakirjan tietosisältöjä, ja tehdä niistä päätelmiä sääntökoneiden avulla.

Asiakirjarakenteiden ontologisoinnissa ilmeni useita teknisiä haasteita, kuten tietokomponenttien järjestyksen ilmaiseminen tai määritysten validointi. Haastavaksi ontologian

muodostamisen tekee RDF-kielen joustavuus, koska RDF mahdollistaa mallintamisen lukemattomilla eri tavoilla. Tutkimuksen aikana muodostettiin useita erilaisia ontologiaversioita sosiaalihuollon asiakastietomallista.

CCTS-menetelmästä muodostettiin meta-tason ontologia, joka sisältää sosiaalihuollossa käytetyt CCTS-mallin mukaiset rakenteet (Kuva 9).



Kuva 9: CCTS-ontologia

CCTS-ontologia on muodostettu CCTS-standardia soveltaen kuvaamaan CCTS-tietokomponenteille määriteltäviä ominaisuuksia. CCTS-ontologia määrittelee seuraavat rakenteet:

- **ccts:CCTS**

  CCTS-ontologian juuriluokka, jossa määritellään kaikille CCTS-luokille yhteisiä ominaisuuksia, kuten kardinaliteetti, määritelmä ja esimerkki.

- **ccts:CC**

  Ydinkomponenttien yliluokka, jonka avulla voidaan määritellä kaikille ydinkomponenteille yhtenäisiä ominaisuuksia ja rajoitteita.

- **ccts:ACC**

  Koosteydinkomponenttia kuvaava luokka. Koosteydinkomponentin sisältämät assosiaatio- ja perusydinkomponentit mallinnetaan owl:Restriction-rajoitteina. Luokan ilmentymä viittaa seuraavaan koosteydinkomponentin sisältämään komponenttiin `ccts:hasNext`-viittauksella.

- **ccts:ASCC**

  Assosiaatioydinkomponenttia kuvaava luokka. Luokan avulla voidaan määritellä assosiaatioydinkomponentille CCTS-spesifikaation mukaiset ominaisuudet ja säilyttää kenttien välinen järjestys.

- **ccts:BCC**

  Perusydinkomponenttia kuvaava luokka. Luokan avulla voidaan määritellä perusydinkomponenteille CCTS-määritelmän mukaiset ominaisuudet ja säilyttää kenttien välinen järjestys.

- **ccts:BIE**

  Toimialakohtaisten tietokomponenttien yliluokka, jonka avulla voidaan määritellä kaikille toimialakohtaisille tietokomponenteille yhtenäisiä ominaisuuksia ja rajoitteita.

- **ccts:ABIE**

  Toimialakohtaista koostetietokomponenttia kuvaava luokka. Luokan ilmentymät ovat aina jonkin `ccts:ACC`-luokan ilmentymien aliluokkia.

- **ccts:ASBIE**

  Toimialakohtaista assosiaatiotietokomponenttia kuvaava luokka. Luokan ilmentymät ovat aina jonkin `ccts:ASCC`-luokan ilmentymien aliluokkia.

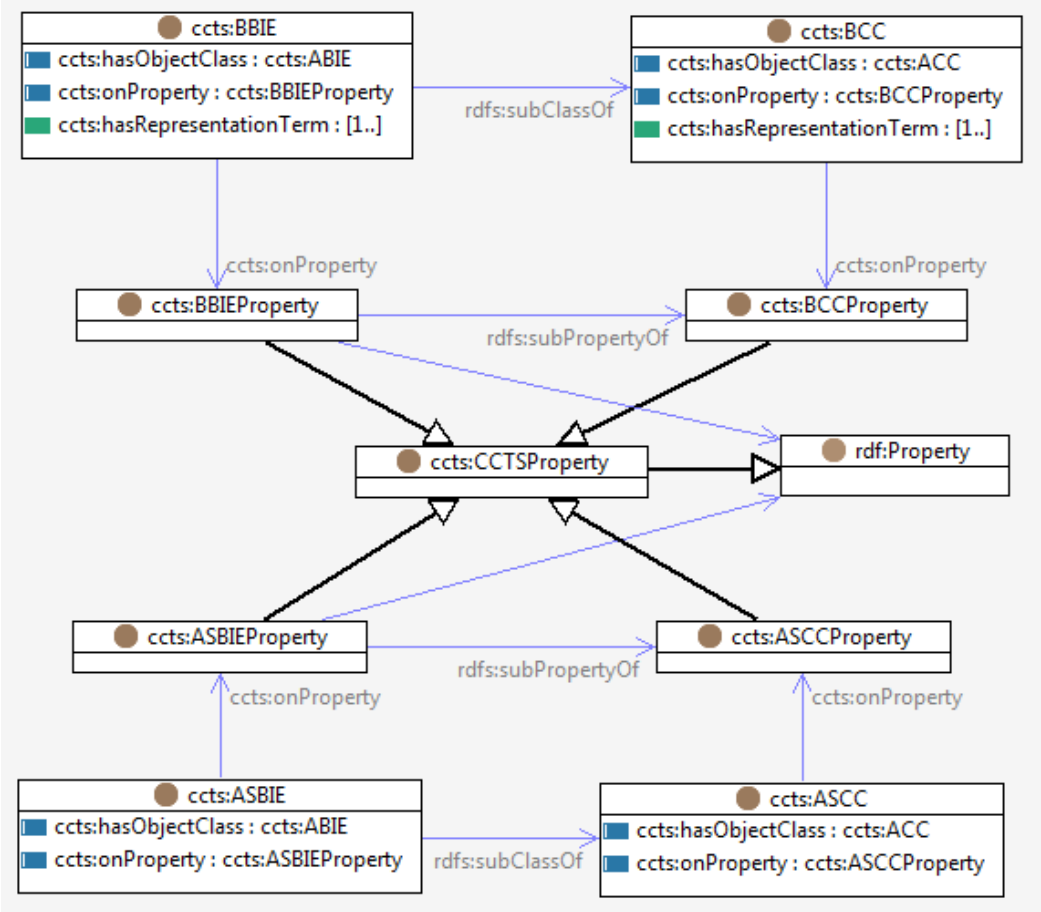
- **ccts:BBIE**

  Toimialakohtaista perustietokomponenttia kuvaava luokka. Luokan ilmentymät on aina jonkin `ccts:BBIE`-luokan ilmentymien aliluokkia.

- **ccts:BD**

  Toimialakohtaista asiakirjarakennetta kuvaava luokka. Luokassa määritellään kaikki asiakirjaan kuuluvat toimialakohtaiset tietokomponentit.

Tietokomponentin kenttien järjestys ontologiassa muodostetaan kenttien välisinä linkeinä `ccts:hasNext`-viittauksen avulla. Assosiaatio- ja perustietokomponenttien suhde uudelleenkäytettävissä olevaan ominaisuuteen (CCTSProperty) on mallinnettu `ccts:onProperty`-viittauksen avulla. CCTS-komponentteja kuvaavien luokkien lisäksi määritellään luokat predikaateille, jotka vastaavat CCTS-määrityksen mukaisia uudelleenkäytettäviä ominaisuuksia (Kuva 10).



Kuva 10: CCTS luokat ja predikaatit

Predikaatit on mallinnettu `rdf:Property`-luokkien aliluokkina. Tietokomponenttien kentistä viitataan predikaatteihin `ccts:onProperty`-viitauksella. Komponenttiluokkien avulla mallinnetaan komponentteihin liittyvät määritykset ja kenttien järjestys. CCTS-ontologiassa on mallinnettu seuraavat predikaatit:

- **ccts:CCTSProperty**

  CCTSProperty-luokka toimii kaikkien CCTS-predikaattien yliluokkana. Se on mallinnettu `rdf:Property`-luokan aliluokaksi. Luokka vastaa abstraktia kuvausta uudelleenkäytettävistä CCTS-ominaisuuksista ja sen avulla voidaan määritellä kaikkia CCTS-ominaisuuksia koskevia sääntöjä.

- **ccts:ASCCProperty**

  Luokan ilmentymien avulla kuvataan koosteydinkomponenttien välisiä suhteita. Luokka on myös `owl:ObjectProperty`-luokan aliluokka.

- **ccts:BCCProperty**

Luokan ilmentymien avulla kuvataan koosteydinkomponentin ja perusydinkomponenttien välisiä suhteita. Luokka on myös `owl:DatatypeProperty`-luokan aliluokka.

- **ccts:ASBIEProperty**

  Luokan ilmentymien avulla kuvataan toimialakohtaisten koostetietokomponenttien välisiä suhteita. Luokan ilmentymien tulee olla `ccts:ASCCProperty`-luokan ilmentyminen aliluokkia.

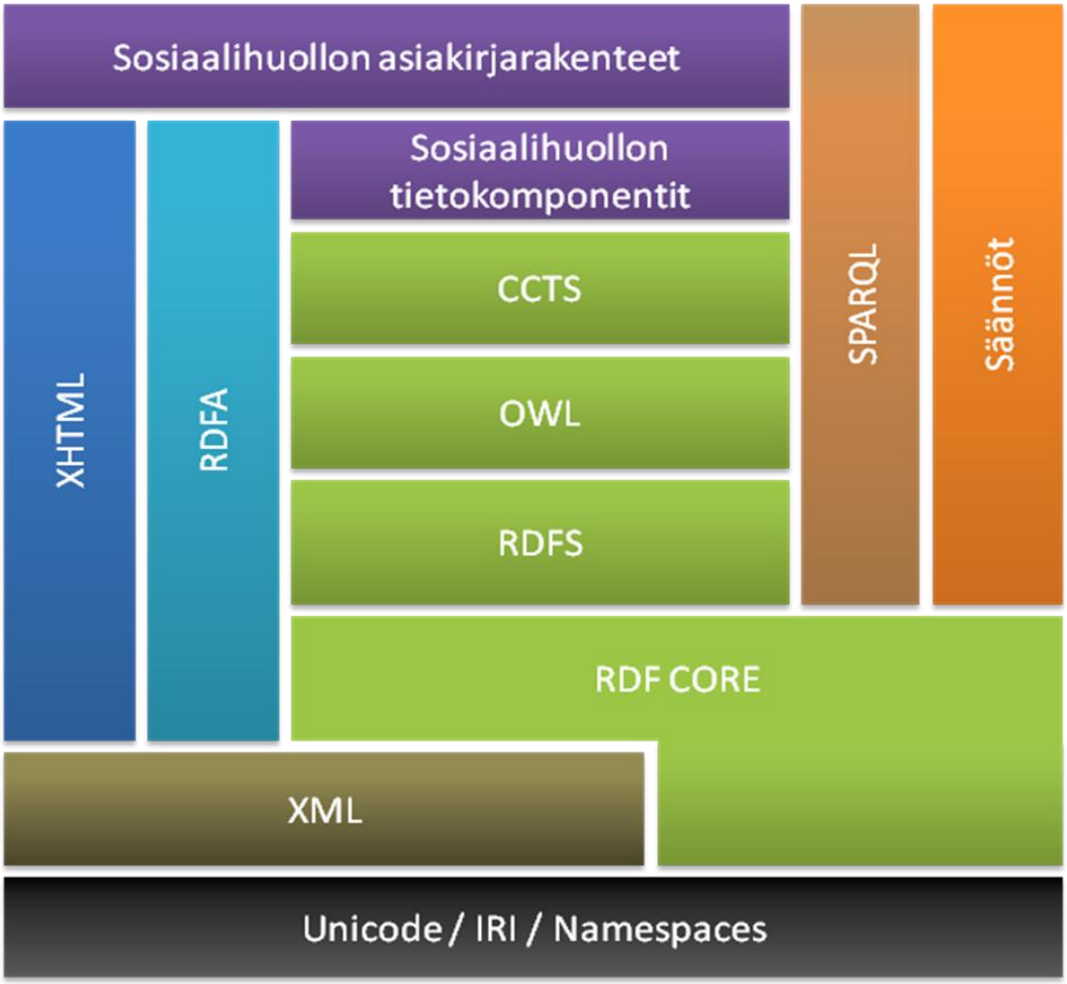
- **ccts:BBIEProperty**

  Luokan ilmentymien avulla kuvataan toimialakohtaisten koostetietokomponenttien ja perusydinkomponenttien välisiä suhteita.

CCTS-ontologia muodostaa perustan tietokomponenttien mallintamiselle ja ylläpidolle ontologiamuodossa. Kun asiakastietomalli ja asiakirjarakenteet on muodostettu määritellyn CCTS-ontologian mukaisesti, on mahdollista generoida XML-skeemat automaattisesti esimerkiksi SPARQL-kyselyn avulla. Asiakirjasisältöjen validointi on myös mahdollista suoraan CCTS-ontologian ja OWL-kielellä määriteltyjen rajoitteiden avulla, kun määritellään ontologiaan pohjautuville ilmentymille selkeät validointisäännöt. CCTS-ontologian mukaisesti annotoitujen XHTML+RDFa asiakirjojen validointisäännöt on kuvattu luvussa 4.

3.3 Sosiaalihuollon asiakastietomallin ontologisointi

Sosiaalihuollon asiakastietomalli ja mallinnetut sosiaalihuollon asiakirjarakenteet kuvattiin Tikesos-hankkeessa taulukoissa. Taulukkopohjaisen asiakastietomallin ylläpito on osoittautunut työlääksi ja hyvin virhealttiiksi menetelmäksi. Sosiaalihuollon asiakastietomallin muuttaminen RDF-muotoon mahdollistaa tietomallin joustavamman ja automaattisen käsittelyn. Sosiaalihuollon semanttinen asiakastietomalli muodostuu RDF-tietomallin mukaisesta CCTS-tietomallista ja sillä määritellyistä tietokomponenteista ja asiakirjoista (Kuva 11).



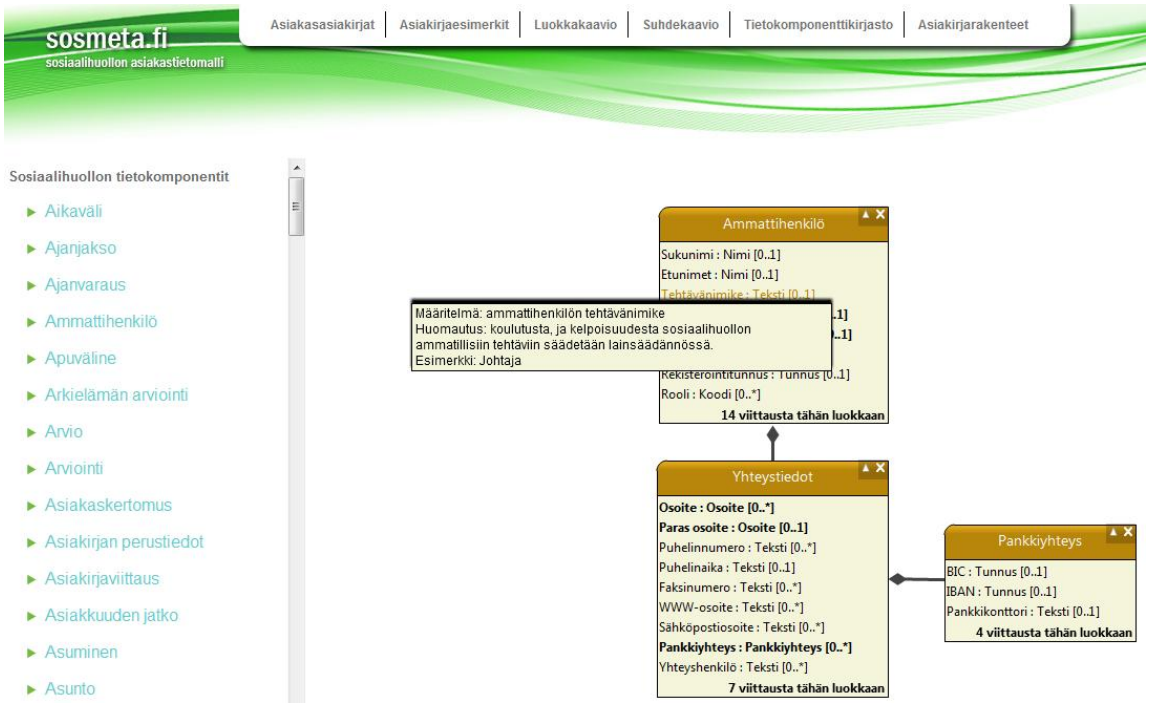
Kuva 11: Sosiaalihuollon asiakastietomallin rakenne

Tutkielmassa on esitetty automaattinen menetelmä taulukkomuotoisten tietokomponenttien ja asiakirjojen muuttamiseen RDF-muotoon. Hankkeessa määritellyt tietokomponenttitaulukot on muutettu ensin CCTS XML -muotoon, josta RDF-tietomalli voidaan generoida automaattisesti XSLT-muunnostiedostolla. XML-muodon generointi taulukoista onnistuu esimerkiksi yksinkertaisten skriptikielten avulla. CCTS XML -muoto on kehitetty helpottamaan siirtymistä taulukkomuotoisista tietokomponenteista muihin esitysmuotoihin, kuten RDF-tietomalleihin. UN/CEFACT kehittää omaa XML For CCTS -standardia [XFC11] tietokomponenttien kuvaamiseen XML-muodossa. Standardin valmistuttua sitä voidaan soveltaa myös sosiaalihuollon tietorakenteiden kuvaamiseen.

Tässä tutkielmassa käytetty CCTS XML -muoto on määritelty Tikesos-hankkeessa tuotettujen tietokomponenttien kuvaamiseen, eikä kuvaa kaikkia CCTS-spesifikaatiossa määriteltyjä rakenteita. CCTS XML –skeema, joka määrittelee tietokomponenttien ku-

vailutiedot, on esitetty liitteessä A. Liitteessä B on kuvattu esimerkki tietokomponentista skeeman mukaisessa muodossa.

Valmiit RDF-muotoiset tietokomponentit ja asiakirjat talletetaan RDF-tietokantaan, joka mahdollistaa niiden keskitetyn ylläpidon ja hallinnan. RDF-tietokannasta voidaan hakea tietoa SPARQL-kyselyrajapinnan kautta. SPARQL-kyselyiden avulla on toteutettu rajapinta tietokomponenttikirjaston ja graafisen tietomallin välillä. Graafinen tietomalli kuvaa sosiaalihuollon tietokomponentteja ja niiden välisiä suhteita helposti luettavassa muodossa (Kuva 12).



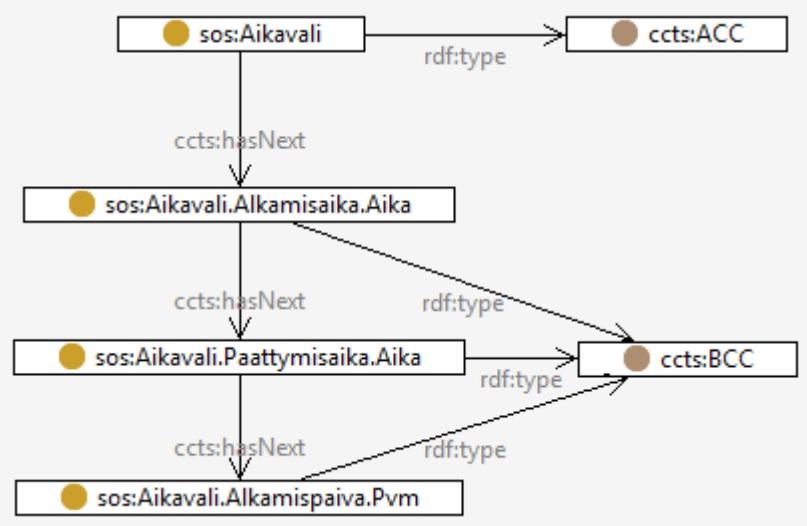
Kuva 12: Graafinen tietomalli

RDF-tietokannan ja CCTS-ontologian pohjalta tietokomponenttien ja asiakirjojen muodostamiseen on mahdollista kehittää metatietorekisteri, jolla tietokomponentteja ja asiakirjoja voidaan kehittää ja ylläpitää. Hollannin oikeusministeriön kehittämän metatietorekisterin soveltuvuutta sosiaalihuollon tietorakenteiden ylläpitoon on tutkittu, mutta sen käyttöönotto edellyttää jatkotoimenpiteitä [AlK11].

3.3.1 Tietokomponenttikirjasto RDF-muodossa

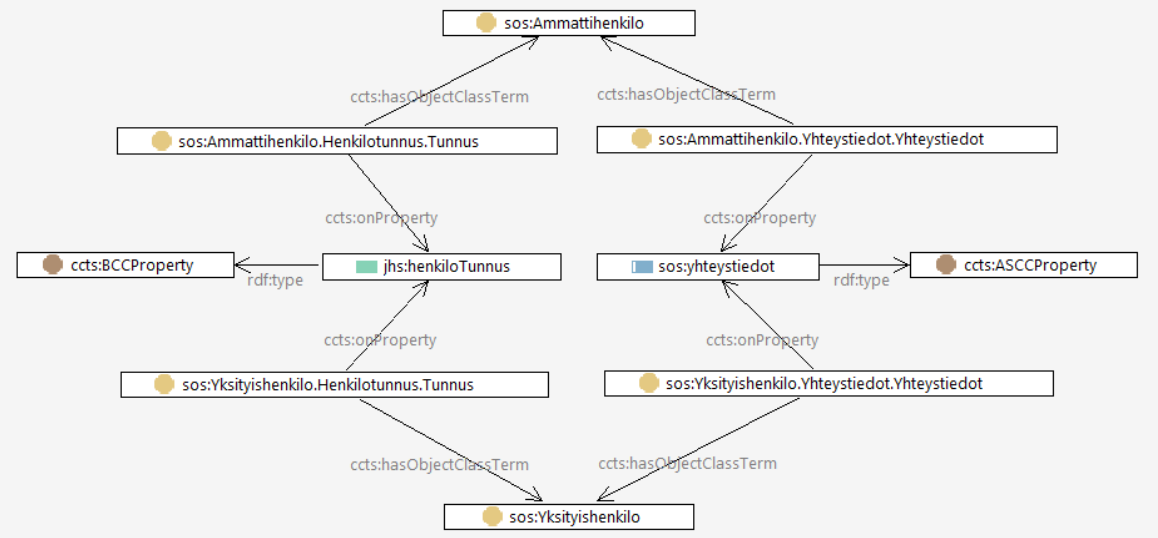
Tietokomponentit mallinnetaan CCTS-ontologiassa määriteltyjen luokkien mukaisesti (Luku 3.2). Ydinkomponentit mallinnetaan `ccts:CoreComponent`-tyyppisiksi luokiksi. Tietokomponenttien kenttien järjestys kuvataan `ccts:hasNext`-viittausten

avulla (Kuva 13). Kenttien järjestyksellä on merkitystä kun asiakastietomallista muodostetaan graafisia tietomalleja tai XML-skeemoja.



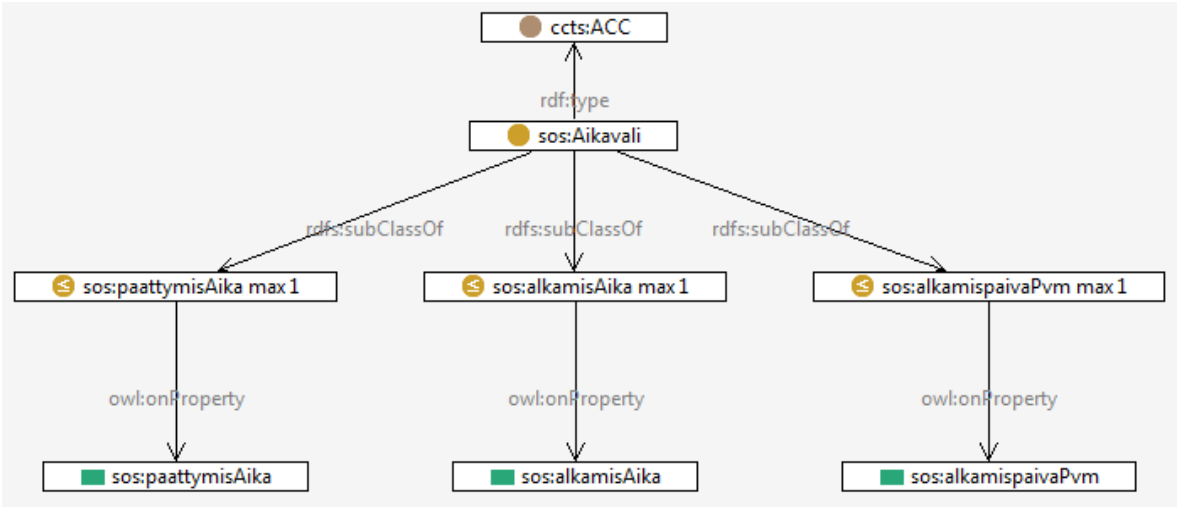
Kuva 13: Tietokomponentin kenttien järjestys

Tietokomponenttien sisältämät kentät ja assosiaatiot muihin luokkiin mallinnetaan OWL-rajoituksina. Assosiaatiot ja tietokentät muodostavat uudelleenkäytettäviä predikaatteja (Kuva 14), jotka toimivat kuten CCTS-spesifikaatiossa määritellyt assosiaatio- ja ominaisuustermit.



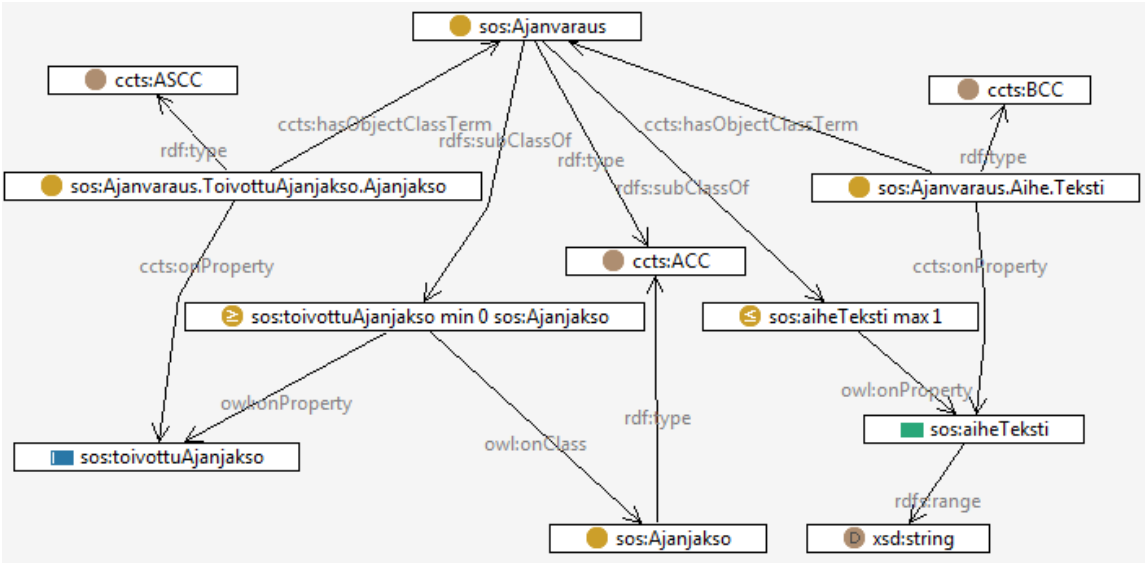
Kuva 14: Uudelleenkäytettävät ominaisuudet

Uudelleenkäytettävien ominaisuuksien avulla luokille määritellään aksioomia, joiden avulla kuvataan toistuvuuksia ja pakollisuuksia (Kuva 15).



Kuva 15: Tietokomponenttien rajoitukset aksioomien avulla

BCC-komponenttien arvoalueet määritellään predikaateissa `rdfs:range`-viittauksen avulla. ASCC-komponenttien viittaukset muihin tietokomponentteihin määritellään `owl:Restriction`-rajoitteissa `owl:onClass`-viittauksen avulla (Kuva 16).

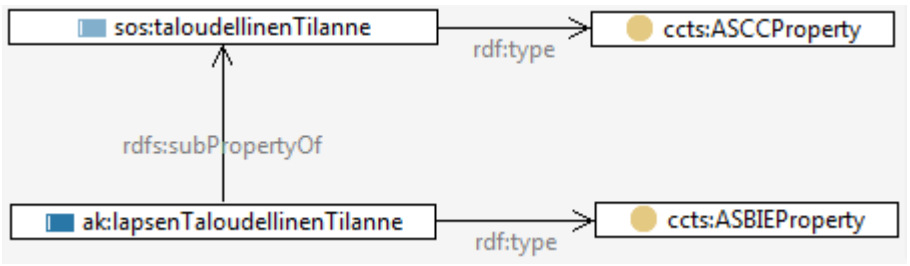


Kuva 16: Arvoalueiden viittaukset

RDF-muotoinen tietokomponenttikirjasto muodostaa sosiaalihuollon asiakastietomallin perustan, jonka pohjalta sosiaalihuollon asiakasasiakirjat muodostetaan. Liitteessä C on XSLT-muunnostiedosto, jonka avulla voidaan muodostaa CCTS-ontologian mukainen tietokomponenttikirjasto CCTS XML -muodosta.

3.3.2 Asiakirjarakenteet RDF-muodossa

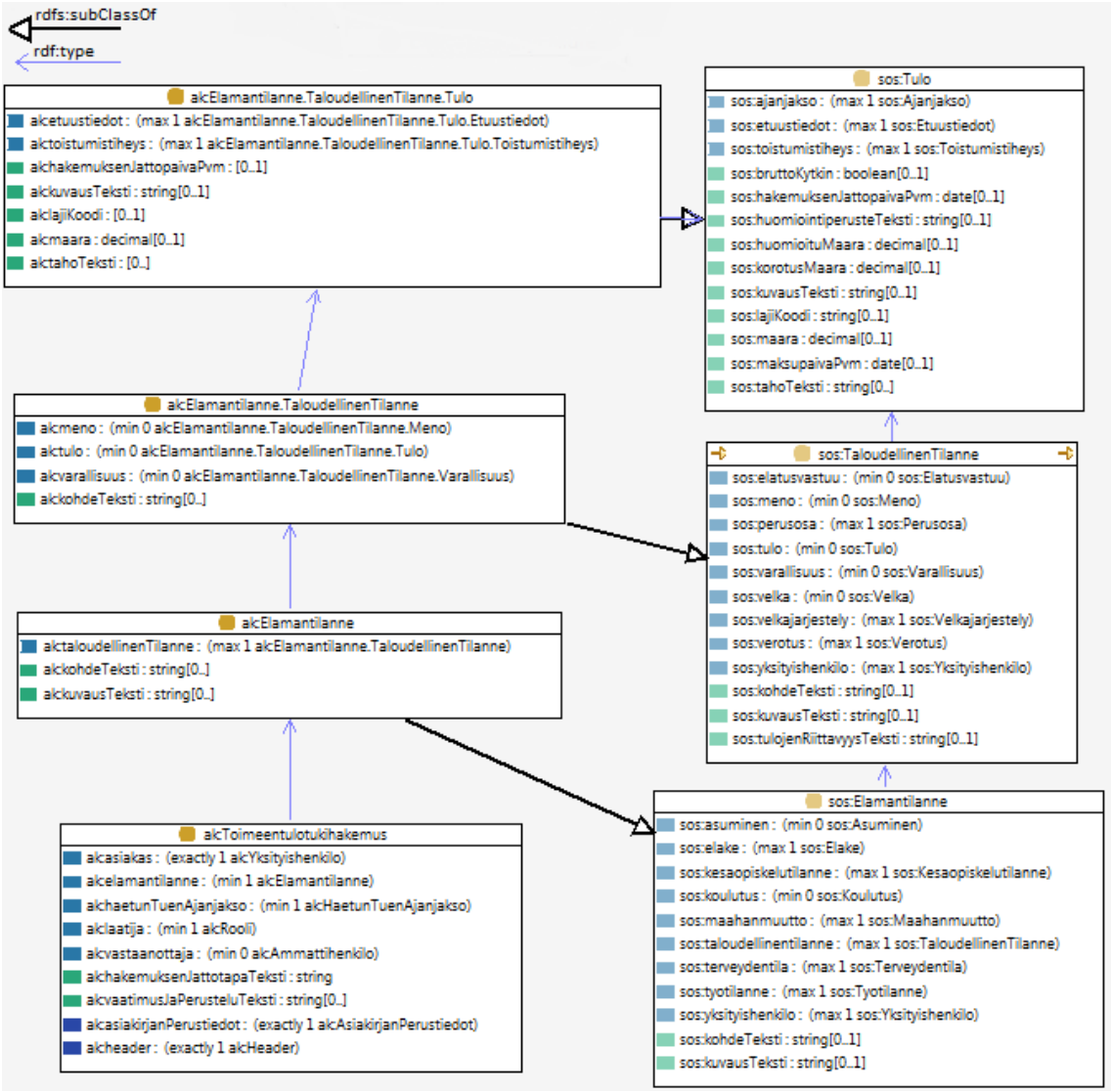
Sosiaalihuollon asiakirjarakenteet muodostetaan erillisiksi RDF-tietomalleiksi, jotka perustuvat tietokomponenttikirjaston tietokomponenteista muodostettuun RDF-tietomalliin. Asiakirjarakenteet mallinnetaan `ccts:BD`-luokan ilmentymiksi. Asiakirjat muodostetaan tietokomponenttikirjastossa olevista tietokomponenteista sekä asiakirjakohtaisista kentistä. Asiakirjamallissa määriteltävien tietokomponenttien tulee perustua sosiaalihuollon asiakastietomallissa määriteltyihin ydintietokomponentteihin. Ydintietokomponenteissa määriteltyjä tietoja voidaan tarkentaa asiakirjakohtaisesti, mutta tarkenteiden tulee kuitenkin aina perustua ydintietokomponenteissa määriteltyihin muuttujiin. Asiakirjakohtaisesti tarkennetut tietokomponentit määritellään tietokomponenttikirjastossa sijaitsevien ydinkomponenttien aliluokiksi (Kuva 17).



Kuva 17: Asiakirjakohtaisten tarkenteiden esittäminen RDF-mallina

Tietokomponentteja voidaan tarpeen mukaan rajoittaa, eli valita vain tarvittavat kentät. Ydinkomponenttien rajoittaminen asiakirjakohtaisesti mahdollistaa rajoitteiden tarkentamisen asiakirjatasolla. Tämä poikkeaa jonkin verran CCTS-menetelmästä, koska menetelmän mukaan asiakirjat rakennetaan kokonaisista ABIE-komponenteista. Toisaalta voidaan katsoa, että asiakirja luo sen kontekstin, jonka sisällä muodostetaan ACC-tietokomponenteista tarkennettuja ABIE versioita.

Kuvassa 18 esitetään yksinkertaistettu versio toimeentulotukihakemuksen asiakirjarakenteesta. Tarkempi XML/RDF-muotoinen tekninen asiakirjarakenne on liitteessä H. Kuvan vasen puoli esittää toimeentulotukihakemus-asiakirjan asiakirjakohtaisia tietokomponentteja, jotka ovat tarkennettuja versioita ydintietokomponenteista.



Kuva 18: Asiakirjakohtaiset tietokomponentit

Asiakirjakohtaisten tietokenttien käyttö on perusteltua, jos kentän tietosisältö on vahvasti sidoksissa asiakirjaan, eikä vastaavaa sisältöä käytetä muissa asiakirjoissa. Asiakirjakohtaiset kentät voidaan esittää CCTS-ontologian avulla `cct s:BBIE`-rakenteiden mukaisesti. Asiakirjassa käytettävät asiakirjakohtaiset kentät määritellään rajoituksilla samalla tavalla kuin tietokomponenteissa käytettävät kentät.

Asiakirjakohtaisesti tarkennettujen tietokomponenttien ja asiakirjakohtaisten kenttien muodostaminen mahdollistaa sosiaalihuollon asiakastietomallin samanaikaisen kehittämisen ja käyttämisen tuotantoympäristössä. Tekniset asiakirjarakenteet eivät ole suoraan riippuvaisia tietokomponenttikirjastosta, joten muutokset ydintietokomponentteihin eivät suoraan vaikuta asiakirjarakenteisiin. Tekniset asiakirjarakenteet voidaan generoida CCTS XML -muodosta, jonka muuntamiseen on muodostettu erillinen XSLT-muunnostiedosto (Liite G).

3.3.3 Sosiaalihuollon asiakirjat XHTML+RDFa-muodossa

Sosiaalihuollon asiakirjat kuvataan XHTML+RDFa muodossa. RDFa-attribuuttien avulla voidaan annotoida XHTML-dokumenttien kohdat, jotka kuvaavat yhteisesti sovittua sosiaalihuollon asiakastietomallia. XHTML+RDFa-suosituksen avulla voidaan muodostaa ihmiselle helppolukuisia asiakirjoja, jotka ovat myös koneellisesti käsiteltävissä. XHTML+RDFa-suosituksen avulla voidaan määritellä XHTML-sivujen sisällön merkitystä ja sivustojen välisten suhteiden semantiikkaa.

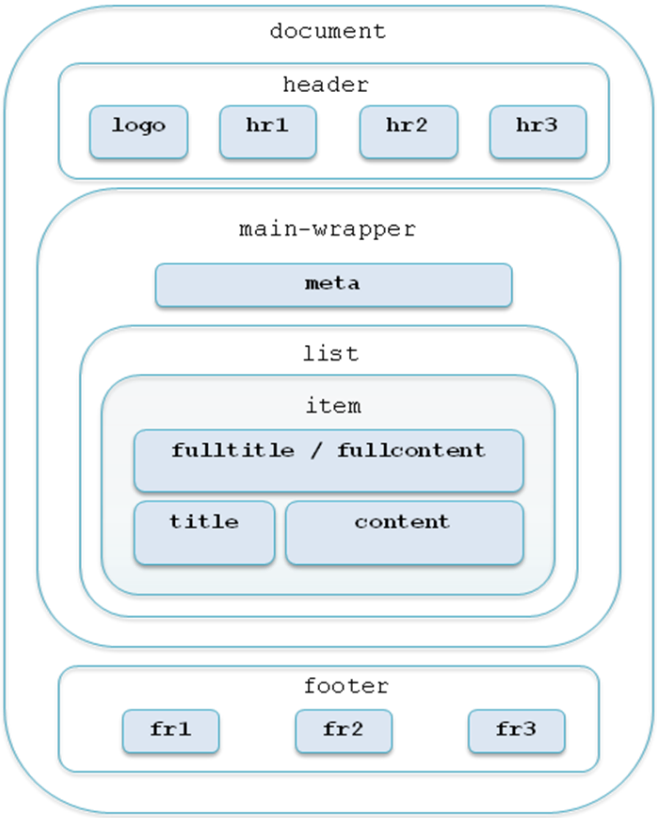
XHTML-asiakirjan rakenne voi perustua yhteisesti sovittuun tai palveluntuottajakohtaisiin asiakirjapohjiin. Tikesos-hankkeessa on tuotettu suositus sosiaalihuollon asiakirjojen näyttömuodosta [AKo11], jota sovelletaan myös XHTML-asiakirjojen CSS-muotoilussa. Tässä luvussa on esitetty XHTML+RDFa-asiakirjapohjan rakenne ja CSS-muotoilu, joiden avulla voidaan kuvata kaikkia sosiaalihuollon asiakirjoja. Esimerkki kokonaisesta asiakirjapohjasta on esitetty liitteessä H. Asiakirjapohja muodostuu sisäkkäisistä div-elementeistä, joiden muotoiluun on määritelty CSS-luokkia:

```
<body>
  <div class="document">
    <div class="header"> ... </div>
      <div class="main-wrapper">
        <div class="main">
          <div class="list">
            <div class="item">
              <div class="title">Otsikko</div>
               <div class="content">Sisältöä</div>
            </div>
         </div>
      </div>
    <div class="footer"> ... </div>
  </div>
</body>
```

Asiakirjapohja ja CSS-luokat mahdollistavat yhdessä asiakirjan rakenteen ja muotoilun. Asiakirjapohjassa määritellään paikat asiakirjan meta- ja tunnistetiedoille sekä asiakirjan tietosisällölle. Yhtenäisen asiakirjapohjan on tarkoitus mahdollistaa XHTML+RDFa-asiakirjojen automaattinen muodostaminen ja tietosisältöjen annotointi.

Kuva 19 havainnollistaa `div`-elementeille CSS-luokkien avulla määriteltyä asettelua. CSS-luokat on määritelty asiakirjapohjassa (Liite H). Asiakirjojen tietosisältö määritellään kuvassa esitettyjen tummennettujen luokkien avulla. Asiakirjan ylä- ja alatunnistetiedot asetellaan asiakirjapohjan `header`- ja `footer`-luokilla. Metatiedot voidaan

määritellä joko XHTML-asiakirjan `head`-elementissä tai asiakirjan muussa rakenteessa meta-luokan avulla. Toistuva asiasisältö määritellään käyttäen `item`-luokkia, joille voidaan määritellä otsikoita ja tietosisältöä `title` ja `content`-luokkien avulla. Asiasisältö voidaan kuvata `content`-luokkien sisällä esimerkiksi tekstimuodossa, taulukkoina tai base64-muodossa.



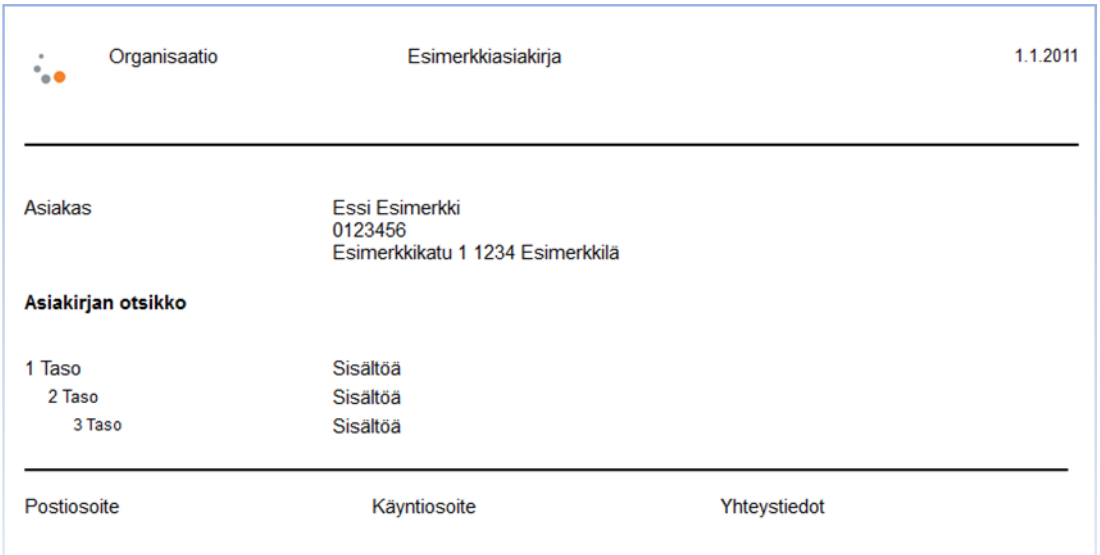
Kuva 19: Asiakirjapohjan asettelu CSS-luokkien avulla

Asiakirjojen semanttinen annotointi aloitetaan `body`-elementistä määrittelemällä asiakirjan tyyppi `typeof`-attribuutin avulla. Asiakirjojen asiasisältö on merkattu toistuvien `item`-luokkien ja `span`-elementtien avulla. `div`-elementillä ja `rel`-attribuutilla kuvataan luokkia ja luokkien välisiä suhteita. `span`-elementillä ja `property`-attribuutilla annotoidaan vapaamuotoisen tekstin sisältöä. Asiakirjarakenteita voidaan yksinkertaistaa jättämällä pois tietokomponenttien `typeof`-luokkamääritykset. Luokkamääritykset voidaan jättää pois, koska käsiteltävästä asiakirjarakenteesta on määritelty asiakirjaskeema, jossa tietokomponenttien väliset suhteet on kuvattu. Seuraavassa esimerkissä on esimerkkiasiakirjan `body`-osa:

```
<body typeof="ak:EsimerkkiAsiakirja">
    <div class="document">
```

```
      <div class="header">
        <div class="logo"></div>
        <div class="hr1"><h1>Organisaatio</h1></div>
        <div class="hr2"><h1>Esimerkkiasiakirja</h1></div>
        <div class="hr3">1.1.2011</div>
      </div>
      <div class="main-wrapper">
         <div class="list">
            <div class="item" rel="sos:asiakasYksityishenkilo">
               <div class="title"><h1>Asiakas</h1></div>
               <div class="content">
                 <p><span property="sos:etuNimi">Essi</span>
                 <span property="sos:sukuNimi">Esimerkki</span></p>
               </div>
               <div class="content" rel="sos:yhteystiedot">
                 <p><span property="sos:puhelinnumeroTeksti">0123456
                 </span><br/></p>
                 <div rel="sos:osoite">
                   <p><span property="sos:osoiteTeksti">
                    Esimerkkikatu 1 1234 Esimerkkilä
                   </span></p>
                 </div>
                </div>
               </div>
             <div class="item">
               <div class="fulltitle"><h1>Asiakirjan
otsikko</h1></div>
              </div>
              <div class="item">
                 <!-- Esimerkkiä on lyhennetty tästä -->
              </div>
          </div>
      </div>
      <div class="footer">
        <div class="fr1"><h1>Postiosoite</h1></div>
        <div class="fr2"><h1>Käyntiosoite</h1></div>
        <div class="fr3"><h1>Yhteystiedot</h1></div>
      </div>
    </div>
  </body>
```

XHTML+RDFa-suosituksen avulla voidaan muodostaa visuaalisesti rikkaita asiakirjoja (Kuva 20), joissa tiedon asettelu ja tekninen rakenne on määritelty samassa tiedostossa.



Kuva 20: Esimerkkiasiakirjan ulkoasu

Sosiaalihuollon asiakasasiakirjan tulee aina muodostaa yksittäinen kokonaisuus, jonka asiasisältö on aina yhdessä tiedostossa. Ulkopuolisiin tietosisältöihin linkittäminen ei ole järkevää asiankäsittelyn ja arkistoinnin näkökulmasta. Kuvat voidaan kuitenkin tallettaa asiakirjaan base64-muodossa, joka mahdollistaa binääridatan koodaamisen ASCII-muotoon.

Asiakirjojen annotoitu tietosisältö ja tekninen rakenne voidaan irrottaa XHTML+RDFa-asiakirjoista RDFa-jäsentimien avulla. RDFa-jäsentimillä voidaan muodostaa useita eri tekstipohjaisia RDF-tiedostomuotoja, kuten JSON ja RDF/XML. Yleisien teksipohjaisten jäsentimien lisäksi on useilla eri ohjelmointikielillä ohjelmoituja jäsentimiä, joiden avulla RDFa-asiakirjojen RDF-sisältöjä voidaan käsitellä olio-pohjaisissa ohjelmistoissa. Seuraavassa esimerkissä on esitetty esimerkkiasiakirjan RDF/XML ABBREV -muoto, joka on jäsennetty edellisestä XHTML+RDFa-esimerkistä:

```
<rdf:RDF>
 <sos:EsimerkkiAsiakirja>
  <sos:asiaSisaltoTeksti
xml:lang="fi">Sisältöä</sos:asiaSisaltoTeksti>
    <sos:asiakasYksityishenkilo>
      <rdf:Description>
        <sos:yhteystiedot>
          <rdf:Description>
            <sos:osoite>
              <rdf:Description>
               <sos:osoiteTeksti xml:lang="fi">
                Esimerkkikatu 1 1234 Esimerkkilä
               </sos:osoiteTeksti>
              </rdf:Description>
            </sos:osoite>
            <sos:puhelinNumero
xml:lang="fi">0123456</sos:puhelinNumero>
```

```
          </rdf:Description>
        </sos:yhteystiedot>
        <sos:etuNimi xml:lang="fi">Essi</sos:etuNimi>
        <sos:sukuNimi xml:lang="fi">Esimerkki</sos:sukuNimi>
      </rdf:Description>
    </sos:asiakasYksityishenkilo>
  </sos:EsimerkkiAsiakirja>
</rdf:RDF>
```

Asiakirjan rakenne voidaan validoida XHTML+RDFa DTD-määrittelyn avulla. Asiakirjojen annotoitujen tietosisältöjen validointiin ei kuitenkaan ole virallista W3C-suositusta. Tutkimuksessa kehitettiin menetelmä tietosisältöjen validointiin sosiaalihuollon asiakastietomallista CCTS-ontologian ja sääntöpohjaisen validoinnin avulla.

4 XHTML+RDFA MUOTOISTEN ASIAKIRJOJEN TIETOSISÄLTÖJEN SÄÄNTÖPOHJAINEN VALIDOINTI

Asiakirjan validoinnilla tarkoitetaan asiakirjailmentymän vertaamista asiakirjasta tehtyyn tietomalliin, joka sisältää asiakirjan rakenteisiin ja sisältöön liittyviä rajoitteita. XML-asiakirjojen validointiin on kehitetty useita ratkaisuja, kuten skeemapohjaiset DTD [SGM86] ja XML Schema [XSD04] sekä sääntöpohjainen Schematron [SCH06].

Skeemapohjainen validointi perustuu asiakirjan rakenteen määrittelyyn erillisenä skeemana, jossa esitetään asiakirjassa käytettävät elementit, elementtien hierarkia ja tietotyypit sekä attribuutit. Sääntöpohjaisessa validoinnissa määritellään sääntöjä tietorakenteiden välisille suhteille. Sääntöpohjaisen validoinnin avulla voidaan yleensä määritellä tarkempia validointisääntöjä kuin skeemapohjaisen validoinnin avulla.

XHTML+RDFa-asiakirjan rakenne voidaan validoida W3C:n DTD-rakennemäärittelyä vastaan [RDA10]. Asiakirjarakenteen validoinnilla varmistetaan, että kaikki asiakirjan rakennemääritelmässä kuvatut elementit ja attribuutit on määritelty, ja että asiakirjassa ei ole muita rakenteellisia virheitä. XHTML+RDFa-asiakirjan DTD-määrittely ei kuitenkaan ota kantaa asiakirjojen RDF-sisältöön.

XML-asiakirjojen validointi perustuu rakenteiden tarkistamiseen muuttumatonta rakennemäärittelyä vasten. RDFS- ja OWL-tietomalleissa määriteltyjä rajoitteita käytetään puolestaan uuden tiedon johtamiseen avoimen oletuksen mukaisesti. Suljetun tietomallin mukaisten rajoitusten [MHS09, GrM05] lisäämistä RDF-tietomalleihin on tutkittu, mutta vastaavia rakenteita ei ole määritelty ontologiakielissä. Suljetun oletuksen mukaisia RDF-tietomalleja on toteutettu käyttäen SPARQL-kyselyitä [Sit09] ja ohjelmallisella käsittelyllä [EYE11], jolla voidaan tarkistaa rakenteita suljetun oletuksen mukaisesti.

XHTML+RDFa-asiakirjoissa annotoitujen tietosisältöjen validointiin RDF-tietomallin avulla ei ole aiemmin määritelty toteutuksia. XHTML+RDFa-standardin soveltuvuutta sosiaalihuollon asiakirjastandardiksi [AKH11] ja RDF-sisällön validointia on tutkittu tämän tutkielman tekemisen aikana [HAK11]. Tässä luvussa on esitetty skeema- ja sääntöpohjaisen validoinnin periaatteita, joiden pohjalta on muodostettu validointisäännöt tutkielmassa esitetyn CCTS-pohjaisen RDF-tietomallin mukaisille asiakirjoille.

Skeemapohjaisen validoinnin periaatteista on tunnistettavissa kolme validointikohdetta, joiden avulla asiakirjojen tietosisällöstä voidaan tunnistaa tyypillisiä virheitä:

- **Kardinaliteetti**
  Sisällön ilmentymille voidaan määritellä minimi- ja maksimilukumäärät. Esimerkiksi voidaan tarkistaa, että yksityishenkilöllä on tarkalleen yksi henkilötunnus.

- **Tietotyypit**
  Sisältöä voidaan verrata tietotyyppeihin ja asettaa tietotyyppien arvoalueille rajoituksia. Esimerkiksi vähäisen tulon määrä voi olla tietotyypiltään desimaaliluku, joka on välillä 0-500.

- **Vastaavuus**
  Sisällölle tulee löytyä vastaavuus tietomallista. Määrittelemätön tietosisältö muodostaa virheen. Esimerkiksi voidaan havaita, että osoitetietoihin on liitetty yksityishenkilön henkilötunnus, joka ei kuulu osoitteen rakenteeseen.

Sääntöpohjaisen validoinnin avulla voidaan asettaa tarkempia rajoituksia asiakirjojen tietosisällöille. Säännöillä voidaan asettaa rajoituksia esimerkiksi seuraaville kohteille:

- **Vaihtoehtoisesti pakolliset kentät**

  Sisällölle voidaan määritellä vaihtoehtoisia kardinaliteetteja. Esimerkiksi jos yksityishenkilö on naimisissa, niin asiakirjassa täytyy olla määritelty myös puolison tiedot.

- **Koodiarvojen tarkastaminen**

  Tarkastetaan koodiarvojen oikeellisuus ulkoisista tai paikallisista koodistoista. Esimerkiksi tarkastetaan, että asiakirjassa oleva kielikoodi löytyy kielikoodistosta.

- **Tietosisällön johdonmukaisuus**

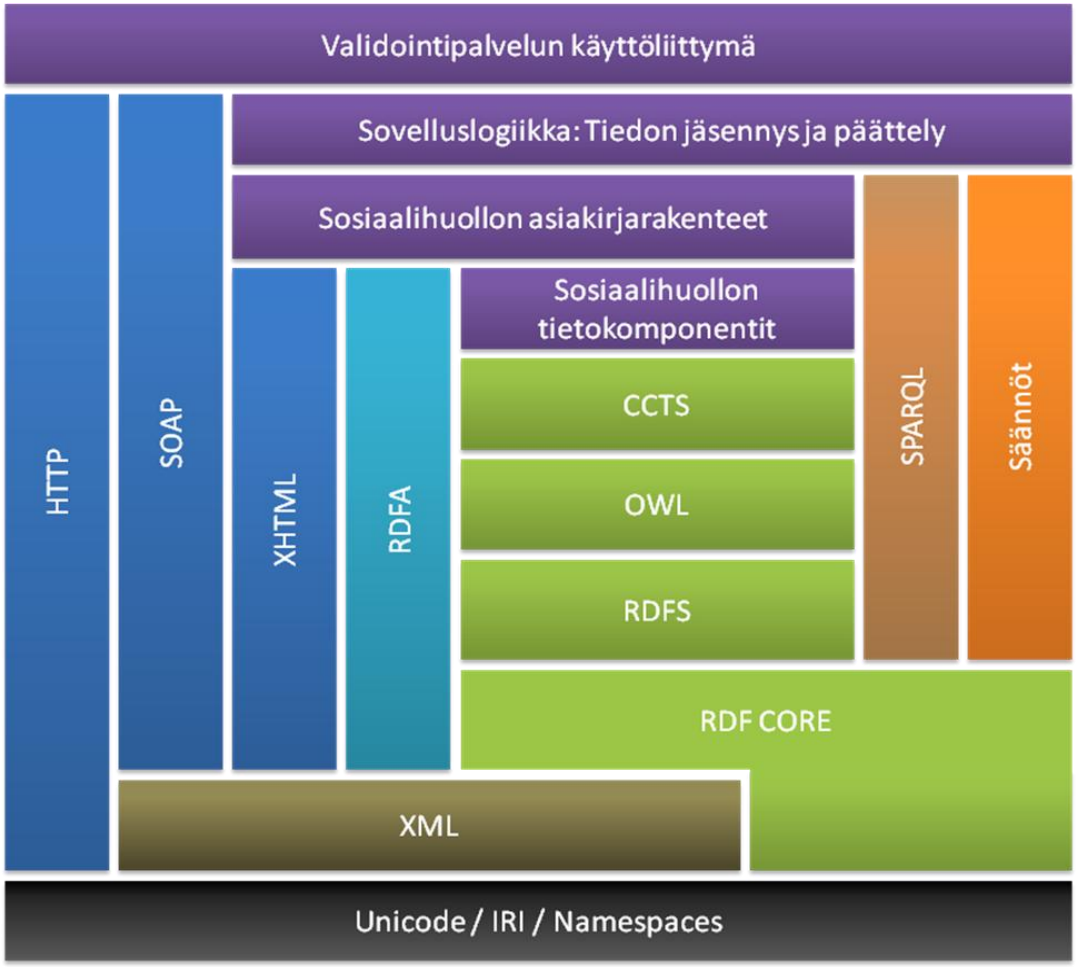
Tarkastetaan sisällön oikeellisuus ja järjestys. Esimerkiksi tarkastetaan, että ajanjakso alkaa ajallisesti ennen sen päättymistä ja että ajanjakson aloittava päivämäärä on esitetty näyttömuodossa ennen ajanjakson päättävää päivämäärää.

- **Toimialakohtaisten sääntöjen muodostaminen**

  Tarkastetaan, että asiakirjan sisältö on toimialakohtaisten sääntöjen mukainen. Toimialakohtaiset säännöt voi olla määritelty palvelutehtävä- tai asiakirjakohtaisesti.

4.1 Validointipalvelun arkkitehtuuri

XHTML+RDFa mahdollistaa toimialakohtaisten asiakirjojen välittämisen eri toimijoiden kesken. XHTML+RDFa-asiakirjojen tietosisältöjen validointiin on tässä tutkimuksessa kehitetty menetelmä, johon on sovellettu semanttisen webin teknologioita (Kuva 21).

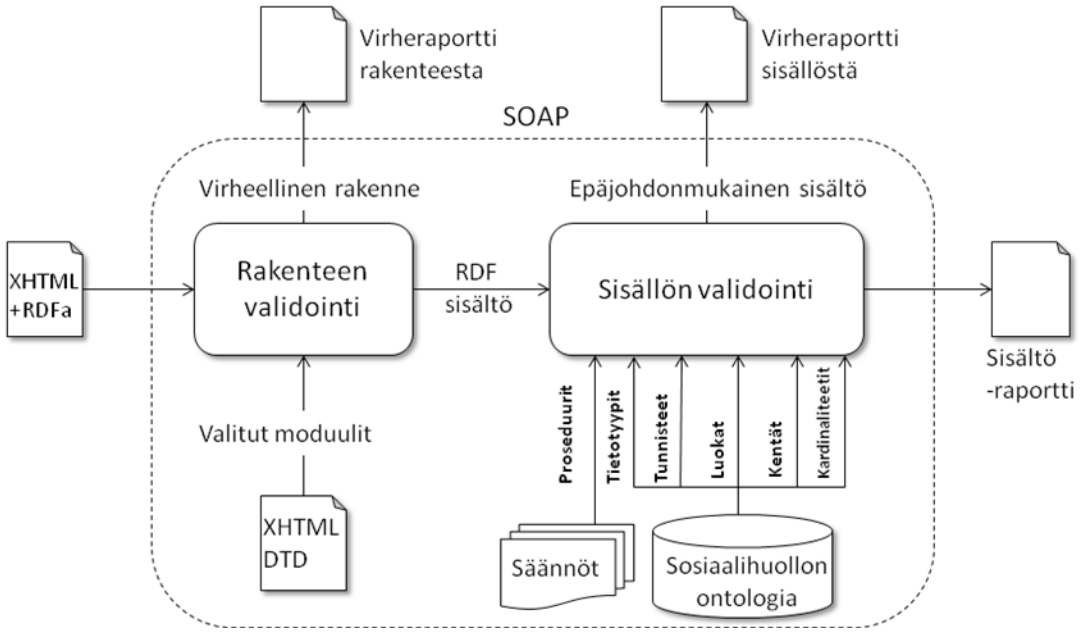


Kuva 21: Validointipalvelun teknologiat

Validointipalvelu on suunniteltu XHTML+RDFa-asiakirjojen rakenteen ja sisällön validointiin. Validointipalvelun toiminnallisuus perustuu CCTS-ontologiaan pohjautuviin sääntöihin ja RDF-tietomallin mukaiseen käsittelyyn. Asiakirjojen validointi voidaan jakaa neljään päävaiheeseen:

1. Validointipalvelu vastaanottaa XHTML+RDFa-asiakirjan
2. XHTML+RDFa-rakenne validoidaan DTD-määritystä vasten RDF-sisällön jäsentämisen yhteydessä
3. RDF-sisältö käsitellään sääntöpohjaisesti ja muodostetaan validointiraportti
4. Validointipalvelu palauttaa validoinnin tulokset

XHTML+RDFa-asiakirjojen validointipalvelu suunniteltiin edellä mainittujen vaiheiden mukaisesti (Kuva 22).



Kuva 22: Semanttinen validointipalvelu

Validointipalvelu saa syötteenä XHTML+RDFa-dokumentin. Ensimmäisessä vaiheessa dokumentin rakenne validoidaan XHTML DTD -määritystä vasten. Validoinnissa voidaan käyttää tarvittaessa vain valittuja XHTML-moduuleja, koska XHTML-suositus määrittelee modulaarisen DTD-kirjaston. Kirjaston avulla voidaan rajoittaa tiettyjen XHTML-elementtien käyttöä [XHM01]. Tiettyjen moduulien käyttöä voidaan rajoittaa, jos ne muodostat tietoturvariskin tai moduulien ei katsota soveltuvan asiakirjallisen tiedon esittämiseen.

XHTML-validoinnin yhteydessä jäsennetään asiakirjan RDF-sisältö. Dokumentin jäsennetty RDF-sisältö yhdistetään asiakirjan rakennetta kuvaavan teknisen asiakirjarakenteen kanssa, joka on esitetty myös RDF-tietomallina (Luku 3.3.2). Tämän jälkeen muodostettua kokonaisuutta käsitellään sääntöpohjaisesti sisällön validointiin kehitetyn validointimallin avulla. Virheellisestä rakenteesta tai epäjohdonmukaisesta sisällöstä tuotetaan virheraportti, jossa nimetään virheiden tyypit ja määritellään ongelmakohtien rivi- ja sarakenumerot.

4.2 Validointipalvelun toteutus

XHTML+RDFa-asiakirjat lähetetään validointipalvelulle SOAP-rajapinnan kautta. Asiakirjat jäsennetään RDF-muotoon RDFa-jäsentimen avulla. Yleisesti käytössä olevat RDFa-jäsentimet eivät talleta tietoa RDFa-attribuutin sijainnista. RDF-sisällön jäsentämiseen toteutettiin RDFa-jäsennin [RDC11], jonka avulla voidaan tallettaa tieto RDFa-attribuutin sijainnista XHTML-rakenteessa. Sijaintitiedot talletetaan jokaiselle asiakirjasta muodostetulle RDF-lausekkeelle. Sijaintitietojen avulla validointipalvelu pystyy ilmoittamaan virheiden sijainnin asiakirjassa.

RDF-sisältö validoidaan sosiaalihuollon asiakastietomallia vasten. Validointipalvelu on toteutettu Jena-sääntökoneen avulla. Jena on Java-pohjainen avoimen lähdekoodin ohjelmistokehys RDF-tietomallin käsittelyyn. Jena sisältää kirjastoja ja rajapintoja RDF-, RDFS- ja OWL-tiedostojen tallentamiseen, muokkaamiseen, kyselyiden suorittamiseen ja sääntöpohjaiseen päättelyyn.

4.2.1 JenaRules

Validointisäännöt on määritelty tekstitiedostoina JenaRules-kielellä [JEN11]. Säännöt haetaan tiedostoista ja luetaan Jenan oliopohjaiseen malliin. Sääntö muodostuu head- ja body-osasta. Head-osassa määritellään joukko faktoja, joiden perusteella voidaan johtaa uutta tietoa body-osassa. Säännöt esitetään RDF-tietomallille tyypillisinä lausekkeina.

```
[esimerkki:
(?s sos:etuNimi "Erkki")
(?s sos:sukuNimi "Esimerkki")
->
(?s sos:kuvausTeksti "Esim Erkki on esimerkki")]
```

Esimerkkisääntö muodostaa uuden lausekkeen, jos käytettävissä olevasta tietomallista löytyy resurssi, jolle on määritelty säännön head-osassa määritellyt lausekkeet. Säännöissä voidaan käyttää myös omia tai Jenan-ohjelmistossa määriteltyjä funktiota, joilla voidaan esimerkiksi vertailla tai laskea resurssien arvoja yhteen. [JEN11]

Validointipalvelun toteutuksessa on käytetty seuraavia funktioita:

- **makeSkolem**(?x, ?v1 … ?vn)

  Muodostaa uuden nimettömän resurssin ?x, johon asetetaan viittaukset resursseihin ?v1 – ?vn. Kaikki samannimiset viitteet resursseista liitetään samaan nimettömään resurssiin. Funktion avulla muodostetaan tarkistusresurssit kaikille asiakirjasta jäsennetyille RDF-lausekkeille.

- **lessThan**(?x, ?y) ja **greaterThan**(?x, ?y)

  Muodostavat totuusarvoja kun ?x < ?y (lessThan) tai ?x > ?y (greaterThan). Arvot voivat olla kokonaislukuja, liukulukuja tai DateTime-tyyppisiä. Funktiota käytetään kardinaliteettien tarkastmisessa asiakirjarakenteista.

- **notEqual**(?x, ?y)

  Muodostaa totuusarvon, jos ?x!=?y. Funktiolla voidaan vertailla literaaleja ja resurssien tunnuksia. Funktion avulla voidaan vertailla asiakirjailmentymän ja skeeman objekteja toisiinsa.

Lisäksi validointipalvelua varten on muodosttu omia funktioita, joiden toiminnallisuutta ei ollut määritelty Jena-ohjelmistokirjastossa:

- **propertyCount(**?s, ?p, ?count**)**

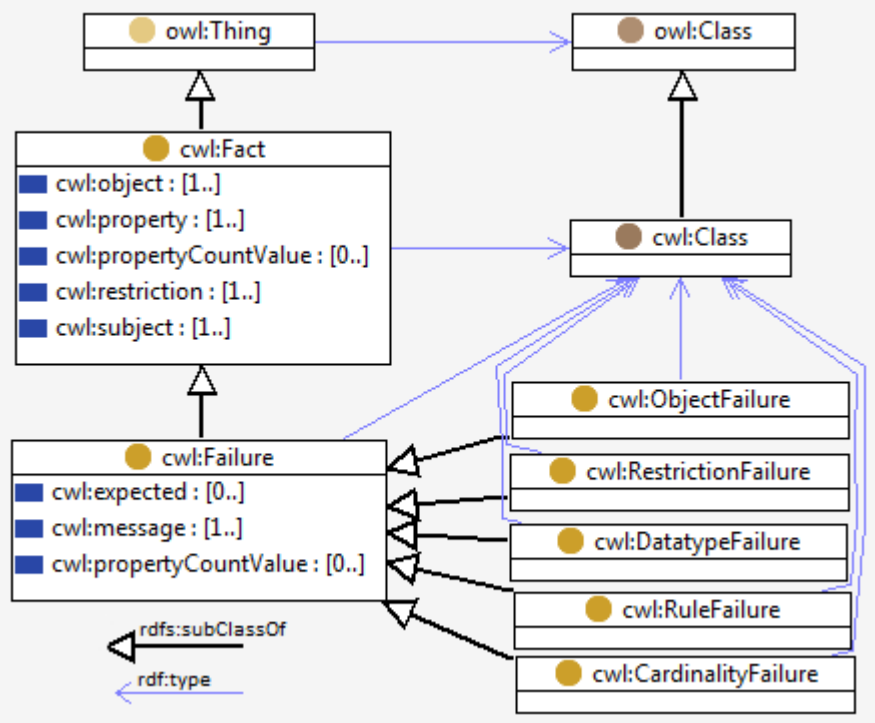
  Laskee kaikki resurssille ?s määritellyt viittaukset ?p ja asettaa viittausten lukumäärän muuttujaan ?count. Funktiota käytetään RDF-lausekkeiden laskemisessa kardinaliteettien tarkastamista varten.

- **notCastableAs(**?x, ?y**)**

Muodostaa totuusarvon, jos arvo `?x` ei ole ymmärrettävissä muodossa `?y`. Eroaa Jenan ohjelmistokirjastossa määritellystä `notDType()`-funktiosta siten, että tarkistettaville arvoille ei tarvitse määritellä tietotyyppiä. Funktiota käytetään tietotyyppien tarkastamisessa.

4.2.2 Validointimalli

Asiakirjan sisällöstä voidaan tunnistaa virheitä RDF-validointia varten suunnitellun validointimallin avulla. CWL-kieli (Closed World Language) on tässä tutkielmassa suunniteltu suljetun oletuksen mukainen (ks. luku 3) tietomalli. CWL lisää RDF/RDFS/OWL-kieliperheeseen uuden tason (Kuva 23), jonka avulla RDF-tietomalleja voidaan tulkita suljetun oletuksen mukaisesti.



Kuva 23: CWL-tietomalli

Uuden tason lisääminen OWL-tietomalliin mahdollistaa RDF-lausekkeiden validoinnin CWL-kielelle määriteltyjen sääntöjen mukaisesti rikkomatta OWL-kielen semanttisia periaatteita. CWL-tietomallissa on määritelty seuraavat luokat:

- **cwl:Class**

  `owl:Class`-luokan aliluokka, joka toimii kaikkien CWL-kielellä mallinnettujen luokkien tyyppinä. Luokan avulla rajoitetaan OWL-luokkien semantiikka suljetun

oletuksen mukaiseksi. CWL-luokkien ilmentymille voidaan määritellä erilliset suljetun oletuksen mukaiset käsittelysäännöt.

- **cwl:Fact**

`owl:Thing`-luokan aliluokka, jonka aliluokat ovat `cwl:Class`-tyyppisiä ilmentymiä. Kaikki `cwl:Class`-luokan ilmentymille määritellyt `owl:Restriction`-rajoitukset validoidaan suljetun oletuksen mukaisesti.

- **cwl:Failure**

`cwl:Fact`-luokan aliluokka, jonka avulla voidaan määritellä puutteellisia luokkia ja luokitella virheellisiä ilmentymiä. `cwl:Failure` voi sisältää useita aliluokkia, jotka tarkentavat virheiden tyyppiä.

4.2.3 Validointisäännöt

Asiakirjojen validointiin on määritelty 9 sääntöä, jotka toteuttavat asiakirjasisältöjen tarkan validoinnin teknisessä asiakirjarakenteessa määriteltyjen rajoitusten mukaisesti. Asiakirjasisällön validointi on toteutettu muodostamalla tarkistusresursseja. Tarkistusresursseja muodostetaan asiakirjan lausekkeille (Taulukko 1) ja asiakirjaskeeman RDF-sisällölle (Taulukko 2). Tarkistusresursseille asetetaan RDF-lausekkeita kuvaavia arvoja CWL-kielen mukaisesti. Asiakirjainstanssin RDF-lausekkeille asetetaan tarkistusresursseissa oletuksena virhetyyppi, jolla ilmaistaan, että kyseinen resurssi on määritelty asiakirjasisällössä.

Taulukko 1

Tarkistusresurssien muodostaminen asiakirjainstanssille

`[CreateInspectorFromInstance:`	Sääntö, joka muodostaa asiakirjainstansseille tarkistusresurssin:
`(?s ?p ?o)`	Käy läpi lausekkeet
`notEqual(?p,rdf:type)`	paitsi luokkaviittaukset
`(?s rdf:type ?class)`	joiden subjektilla on luokka
`(?class rdf:type ?metaClass)`	jolle on määritelty metaluokka
`(?metaClass rdfs:subClassOf cct:BIE)`	joka on ccts:BIE-luokan aliluokka
`makeSkolem(?inspector,?s,?p)`	Määrittele kokonaisuudelle tarkistusresurssi
`->`	
`(?inspector cwl:subject ?s)`	Liitä resurssiin subjekti
`(?inspector cwl:property ?p)`	Liitä resurssiin ominaisuus
`(?inspector cwl:object ?o)`	Liitä resurssiin objekti
`(?inspector cwl:subjectClass ?class)`	Liitä resurssiin subjektin luokka
`(?inspector rdf:type cwl:RestrictionFailure)]`	Määrittele tarkistusresurssin tyypiksi virhe

Tarkistusresurssit muodostetaan `makeSkolem()`-funktiolla, jonka avulla samaan tarkistusresurssiin liitetään asiakirjainstanssin lausekkeet ja skeemoissa määritellyt vastaavuudet. Asiakirjaskeemasta muodostettaviin tarkistusresursseihin (Taulukko 2) liitetään linkki skeemassa määriteltävään rajoitukseen. Rajoitusten ja puutteellisten asiakirjasisältöjen pohjalta sääntökone muodostaa tarkistusresursseista virheitä, jotka voidaan paikantaa XHTML+RDFa-rakenteesta rivi- ja sarakenumeroiden avulla.

Taulukko 2

Tarkistusresurssien muodostaminen skeemalle

`[CreateInspectorFromSchema:`	Sääntö, joka muodostaa asiakirjaskeeman lausekkeista tarkistusresurssit:
`(?s rdf:type ?class)`	Käy läpi subjektit joilla on luokka
`(?class rdf:type ?metaClass)`	jolle on määritelty metaluokka
`(?metaClass rdfs:subClassOf ccts:BIE)`	joka on ccts:BIE aliluokka
`(?class rdfs:subClassOf ?restriction)`	ja joille on määritelty rajoitus
`(?restriction rdf:type owl:Restriction)`	jonka on tyyppi owl:Restriction
`(?restriction owl:onProperty ?p)`	ja sille on määritelty ominaisuus
`makeSkolem(?inspector,?s,?p)`	Määrittele kokonaisuudelle tarkistusresurssi
`->`	->
`(?inspector cwl:subject ?s)`	Liitä resurssiin subjekti
`(?inspector cwl:property ?p)`	Liitä resurssiin ominaisuus
`(?inspector cwl:subjectClass ?class)`	Liitä resurssiin subjektin luokka
`(?inspector cwl:restriction ?restriction) ]`	Liitä resurssiin skeemassa määritelty rajoite

Kaikista tarkistusresursseista käydään läpi ne, joista löytyy linkki skeemassa määriteltyyn rajoitukseen ja asiakirjasisällön perusteella muodostettu virhetyyppi (Taulukko 3). Kun virhetyyppi poistetaan näistä resursseista, jäljelle jäävät tarkistusresurssit edustavat niitä lausekkeita, joita ei ole määritelty asiakirjaskeemasta. Näin voidaan tarkistaa, että asiakirjassa käytetään vain skeemassa määriteltyjä rakenteita.

Taulukko 3

Tarkistusresurssien tarkistaminen

`[ClearSchemaInspector:`	Sääntö, joka käy skeemaan perustuvat lausekkeet
`(?inspector rdf:type cwl:RestrictionFailure)`	jos tarkistusresurssi on määritelty virheeksi
`(?inspector cwl:restriction ?restriction)`	ja tarkistusresurssi perustuu skeemaan
`->`	
`drop(0) ]`	poista virhe tarkistusresurssista

Tämän jälkeen tarkistetaan, että asiakirjassa määritellyt arvot (Taulukko 4) ja viittaukset luokkiin (Taulukko 5) on määritelty skeeman mukaisesti. Tietotyyppien tarkistamiseen on käytetty `notCastableAs()`-funktiota, joka palauttaa totuusarvon, jos annettu arvo ja tietotyyppi eivät ole yhteensopivia.

Taulukko 4

Tietotyyppien tarkistaminen

`[DatatypePropertyRangeError:`	Sääntö, jolla luodaan tietotyyppivirheet
`(?inspector cwl:property ?p)`	Jos on olemassa resurssi, joka viittaa predikaattiin
`(?p rdf:type ccts:BBIEProperty)`	joka on tyyppiä BBIEProperty
`(?inspector cwl:object ?o)`	jolle on määritelty tietotyyppi
`(?p rdfs:range ?range)`	ja predikaatti viittaa objektiin
`notCastableAs(?o, ?range)`	eikä objekti ole tietotyypin mukainen
`->`	
`(?inspector rdf:type cwl:DatatypeFailure)`	Aseta resurssin tyypiksi tietotyyppivirhe
`(?inspector cwl:expected ?range)]`	Aseta resurssille viite odotettuun tietotyyppiin

Tietokomponenttien väliset objektiviittaukset tarkistetaan skeemassa määriteltyjen yksilöllisten URI-tunnisteiden avulla. Asiakirjassa käytettyjen viittausten URI-tunnisteita verrataan skeemassa määriteltyihin URI-tunnisteisiin notEqual-funktion avulla, joka palauttaa totuusarvon, jos URI-tunnisteet eivät ole samoja.

Taulukko 5

Objektiviittausten tarkastaminen

`[ObjectRangeError:`	Sääntö, jolla luodaan viittausvirheet
`(?inspector cwl:property ?p)`	Jos löytyy resurssi, jolle on määritelty viittaus
`(?p rdf:type ccts:ASBIEProperty)`	joka on objektiviittaus
`(?inspector cwl:object ?o)`	ja viitteelle on määritelty kohde
`(?o rdf:type ?oType)`	jolla on tyyppi
`(?p rdfs:range ?range)`	ja viittaukselle on määritelty arvoalue
`notEqual(?oType, ?range)`	jos viittauksen kohde ja arvoalue eivät täsmää
`->`	
`(?inspector rdf:type cwl:ObjectFailure)`	Muodosta tarkistusresurssille virhetyyppi
`(?inspector cwl:expected ?range)]`	Lisää tarkistusresurssiin odotettu arvoalue

Kaikki asiakirjassa käytetyt viittaukset lasketaan ja viittausten lukumäärät asetetaan tarkistusresursseille (Taulukko 6). Viittausten lukumäärän laskeminen on toteutettu `propertyCount()`-funktion avulla.

Taulukko 6

Viittausten laskeminen

`[CountProperties:`	Sääntö, joka laskee viittausten määrän
`(?inspector cwl:property ?p)`	Jos löytyy resurssi, jolle on määritelty predikaatti
`(?inspector cwl:subject ?s)`	ja resurssilla on subjekti
`(?inspector cwl:restriction ?restriction)`	ja resurssi on määritelty skeemassa
`(?restriction owl:onProperty ?p)`	jossa on määritelty predikaatti
`propertyCount(?s, ?p, ?count)`	Laske kaikkien viitteiden lukumäärä subjektista
`->`	
`(?inspector cwl:propertyCountValue ?count)]`	Aseta viittausten lukumäärä resurssiin

Asiakirjassa käytettyjen viittausten lukumäärien avulla tarkistetaan noudattaako asiakirja skeemassa määriteltyjä kardinaliteetteja. Asiakirjaskeemassa on määritelty luokkien ominaisuuksille minimi, maksimi ja eksakteja lukumääriä. Viittausten maksimilukumäärä tarkistetaan `greaterThan()`-funktion avulla (Taulukko 7).

Taulukko 7

Maksimilukumäärän tarkastaminen

`[CheckMaxCardinality:`	Sääntö, joka muodostaa virheen, jos kenttiä on määritelty liikaa
`(?inspector cwl:restriction ?restriction)`	Jos löytyy resurssi, jolle on määritelty rajoite
`(?inspector cwl:propertyCountValue ?count)`	ja resurssille on määritelty viittausten lukumäärä
`(?restriction owl:maxCardinality ?maxCardinality)`	ja rajoituksen tyyppi on maxCardinality
`greaterThan(?count, ?maxCardinality)`	ja viittausten lukumäärä on suurempi kuin maxCardinality
`->`	->
`drop(0)`	Poistaa rajoituksen, jotta lausekkeita ei tutkita enää uudestaan
`(?inspector rdf:type cwl:CardinalityFailure)`	Määrittele resurssille virhetyyppi
`(?inspector cwl:message "maxCardinality failure")`	Määrittele resurssille virhearvo
`(?inspector cwl:expected ?maxCardinality)]`	Määrittele resurssille viitteiden odotettu maksimilukumäärä

Asiakirjoissa esiintyvien viittausten minimilukumäärät tarkistetaan `lessThan()`-funktion avulla (Taulukko 8).

Taulukko 8

Minimilukumäärän tarkastaminen

`[CheckMinCardinality:`	Sääntö, joka muodostaa virheen jos kenttiä on liian vähän
`(?inspector cwl:restriction ?restriction)`	Jos löytyy resurssi jolle on määritelty rajoite
`(?inspector cwl:propertyCountValue ?count)`	ja resurssille on määritelty viittausten lukumäärä
`(?restriction owl:minCardinality ?minCardinality)`	ja rajoituksen tyyppi on minCardinality
`lessThan(?count, ?minCardinality)`	ja viittausten lukumäärä on pienempi kuin minCardinality
`->`	->
`drop(0)`	Poistaa rajoituksen, jotta lausekkeita ei tutkita enää uudestaan
`(?inspector rdf:type cwl:CardinalityFailure)`	Määrittele resurssille virhetyyppi
`(?inspector cwl:message "minCardinality failure")`	Määrittele resurssille virhearvo
`(?inspector cwl:expected ?minCardinality)]`	Määrittele resurssille viitteiden odotettu enimmäislukumäärä

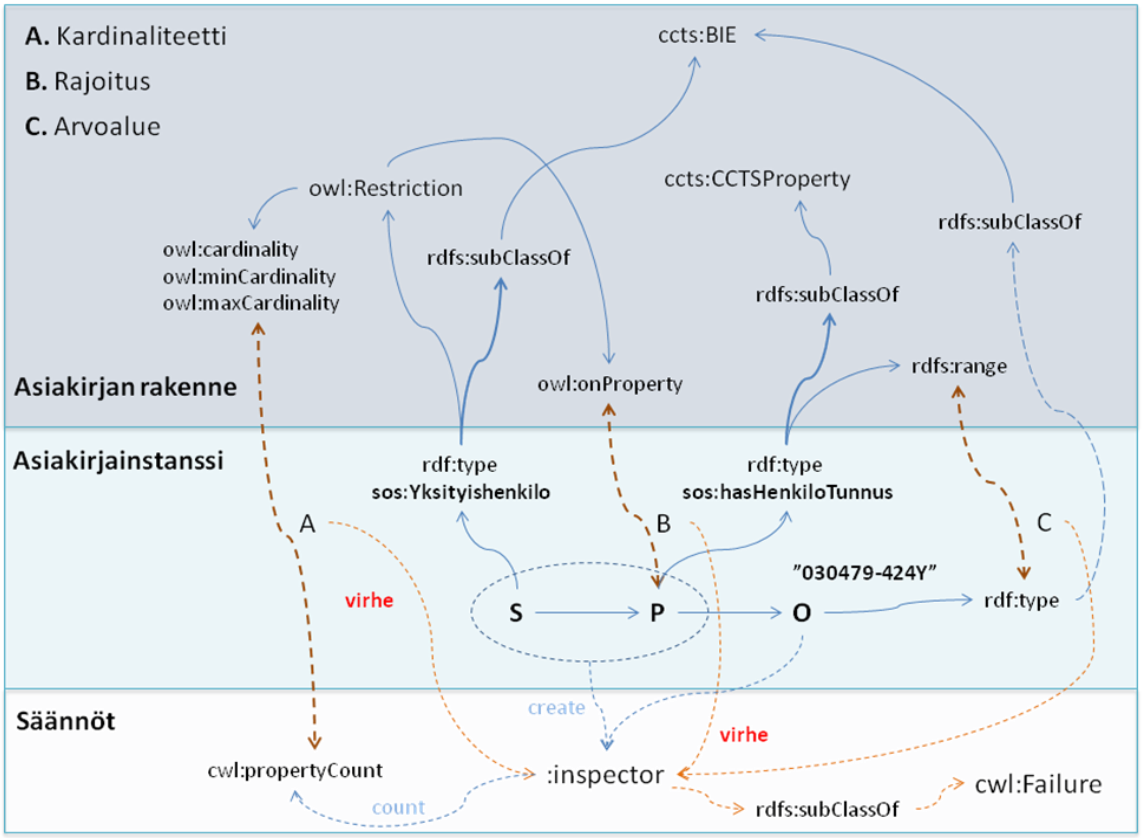
Asiakirjoissa käytettävien viitteiden eksaktien määrittely tarkistetaan `notEqual()`-funktion avulla (Taulukko 9).

Taulukko 9

Tarkan kardinaliteetin tarkastaminen

`[CheckExactCardinality:`	Sääntö, joka muodostaa virheen jos kenttiä oikea määrä
`(?inspector cwl:restriction ?restriction)`	Jos löytyy resurssi, jolle on määritelty rajoite
`(?inspector cwl:propertyCountValue ?count)`	ja resurssille on määritelty viittausten lukumäärä
`(?restriction owl:cardinality ?cardinality)`	ja rajoituksen tyyppi on cardinality
`notEqual(?count, ?cardinality)`	ja viittausten lukumäärä on sama kuin cardinality
`->`	->
`drop(0)`	Poista rajoitus
`(?inspector rdf:type cwl:CardinalityFailure)`	Määrittele resurssille virhetyyppi
`(?inspector cwl:message "cardinality failure")`	Määrittele resurssille virhearvo
`(?inspector cwl:expected ?cardinality)]`	Määrittele resurssille viitteiden odotettu enimmäislukumäärä

Sääntökone käy läpi kaikki asiakirjainstanssin lausekkeet ja käsittelee niitä validointisääntöjen (Taulukko 1-9) mukaisesti. Validointisäännöt muodostavat asiakirjainstanssin lausekkeista ja validointimallista uusia resursseja validointimalliin. Kuva 24 esittää esimerkkiä yhdestä asiakirjasta jäsennetystä RDF-lausekkeesta, johon kohdistetaan validointisääntöjä.



Kuva 24: Lausekkeen validointi

Yhteen asiakirjainstanssin lausekkeeseen liittyy aina useita asiakirjan rakenteessa ja validointimallissa määriteltyjä lausekkeita. Kuvassa on esitetty kolme validointipistettä: kardinaliteetti, rajoitus ja arvoalue, joissa asiakirjainstanssin ja skeeman arvojen tulisi vastata toisiaan.

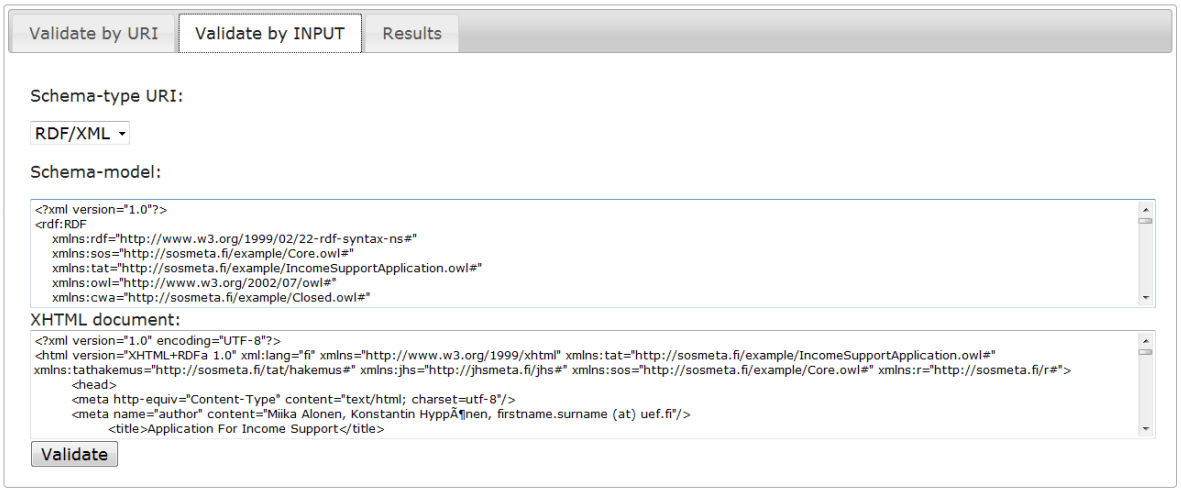
Kardinaliteettien vastaavuuden määrittely (Kohta A) on toteutettu laskemalla ensin asiakirjan lausekkeiden lukumäärä `propertyCount()`-funktion avulla. Tämän jälkeen laskettua lukumäärää verrataan asiakastietomallissa määriteltyyn kardinaliteettiin. Jos lukumäärät eivät vastaa toisiaan, muodostetaan virhetyyppi määriteltyjen sääntöjen mukaisesti.

Asiakirjainstanssin lausekkeiden vastaavuus asiakirjan rajoitteisiin (Kohta B) voidaan todentaa muodostamalla yhteiset tarkistusresurssit sekä asiakirjan rakenteesta, että asiakirjainstanssista. Yhteisistä tarkistusresursseista voidaan erotella ne, jotka edustavat lausekkeita, joille ei ole määritelty rajoituksia asiakirjan tietomallissa. Rajoitteiden avulla voidaan varmistaa, että asiakirjainstanssissa ei ole tehty kirjoitusvirheitä rakennekuvauksissa, tai että rakenteita ei ole liitetty vääriin luokkiin.

Asiakirjainstanssissa esitettyjen arvojen oikeellisuus (Kohta C) todennetaan vertaamalla lausekkeiden todellisia arvoja ja asiakirjan tietomallissa määriteltyjä tietotyyppejä tai luokkia. Arvoalueiden tarkistamisella varmistutaan siitä, että asiakirjainstanssissa määritellyt arvot ovat ymmärrettävissä asiakirjan tietomallissa määriteltyjen tietotyyppien ja luokkien mukaisesti.

4.2.4 Validointipalvelun esimerkkitoteutus

Validointipalvelusta on tehty esimerkkitoteutus (Kuva 25), jonka avulla tutkielmassa muodostettua asiakirjojen kuvausmenetelmää ja asiakirjojen validointia on testattu oikeassa käyttöympäristössä [VAL11]. Validointipalvelun käyttöliittymä on toteutettu yksinkertaisilla JQuery-komponenteilla [JQR11] ja XMLHTTPRequest-objektilla [XHO11], joka mahdollistaa HTTP-kutsujen lähettämisen ja vastaanottamisen JavaScript-kielellä. Esimerkkitoteutuksen avulla on mahdollista testata asiakirjoja asiakastietomallin suunnittelu- ja käyttöönottovaiheessa. Validointipalvelulla voidaan tarkistaa XHTML+RDFa-asiakirjojen tietosisältöjä suoraan tekstisyötteenä tai viittaamalla asiakirjainstanssiin ja skeemaan URL-osoitteen avulla.



Kuva 25: Validointipalvelun toteutus

Validointipalvelun prototyypin toiminnallisuus perustuu edellisessä kappaleessa määriteltyihin sääntöihin. Validointipalvelu mahdollistaa hyvin joustavien tai tiukkojen validointisäännöstöjen luomisen. Validointipalvelulle määriteltyjä sääntöjä voidaan tarvittaessa myös supistaa tai laajentaa eri käyttötarkoituksia varten.

5 POHDINTA

Tässä tutkielmassa on esitetty RDF-tietomalliin pohjautuva menetelmä sosiaalihuollon asiakastietomallin muodostamiseen ja käsittelyyn. Sosiaalihuollon tietokokonaisuudet voidaan mallintaa tutkielmassa esitetyn RDF-tietomallin avulla ja toteuttaa asiakirjat XHTML+RDFa-suosituksen mukaisesti. Tutkielmassa esitetty sääntöpohjainen validointipalvelu mahdollistaa XHTML+RDFa-asiakirjojen rakenteellisen ja sisällöllisen validoinnin.

Asiakirjat ja viestisanomat toteutetaan usein XML-skeemojen ja skeemakirjastojen avulla. Asiakirjoja koostetaan yleensä muodostamalla useasti toistuvista tietokokonaisuuksista uudelleenkäytettäviä tietokomponentteja skeemakirjastoihin. XML-muoto soveltuu hyvin tiedonsiirtoon, ja asiakirjojen validointi on yksinkertaista XML-skeemaa vasten. XML-muotoinen asiakirja vastaa kuitenkin enemmän sanomarakennetta kuin varsinaista asiakirjaa ja ulkoasua. XML-rakenteen lisäksi täytyy määritellä erilliset näyttömuototiedostot, jolloin osa asiakirjan informaatiosta on erillisissä tiedostoissa.

Tietokomponentteja ja asiakirjoja voidaan mallintaa muodostamalla UML-kaavioita tai taulukoita. Asiakirjaskeemojen ja ydinkomponenttiskeemojen ylläpito kuitenkin vaikeutuu huomattavasti asiakirjojen määrän kasvaessa. Asiakirjojen ylläpitäminen erillisissä mallinnusmuodoissa aiheuttaa päivitystarpeita useisiin erillisiin tietomäärityksiin, koska eri mallinnusmuodot eivät ole yhteydessä toisiinsa. Päivitykset tietokomponentteihin aiheuttavat päivityksiä asiakirjaskeemoihin, näyttömuototiedostoihin ja graafisiin tietomalleihin. Kun asiakirjakohtaisten skeemojen lukumäärä on suuri, esimerkiksi sosiaalihuollossa 230 skeemaa ja saman verran näyttömuototiedostoja, kokonaisuuden ylläpito käy haasteelliseksi.

XHTML+RDFa mahdollistaa asiakirjarakenteen ja näyttömuodon toteutuksen yhteen tiedostoon, jonka rakenne voidaan validoida yhden XML-skeeman tai DTD-määrittelyn avulla. XHTML+RDFa on kansainvälinen standardi, jonka käyttämiseen on paljon informaatiota ja valmiiksi toteutettuja ohjelmistokirjastoja asiakirjarakenteiden käsittelyyn. XHTML mahdollistaa lomakkeiden ja asiakirjojen toteuttamisen kevyiden nettikäyttöliittymien avulla.

XML Schema -pohjainen tietomalli ja tutkielmassa esitelty XHTML+RDFa-ratkaisu eivät eroa paljoa toisistaan, jos asiaa mietitään asiakirjojen muodostajan näkökulmasta. Molempien tietomallien pohjalta muodostetaan XML-pohjaisia asiakirjoja, joissa on ennalta määrätyt paikat asiakastiedoille. XHTML-asiakirjojen sisällön validointi edellyttää kuitenkin erillistä validoinnin toteutusta XML-skeemaan verrattuna. XHTML+RDFa-asiakirjojen sisällön validointiin on esitetty tutkielmassa ratkaisu, jonka avulla voidaan varmistua siitä, että XHTML-rakenteet ja sisältö on oikein muodostettu.

Asiakastietomallin ylläpitäjän näkökulmasta RDF-tietomalli mahdollistaa asiakirjarakenteiden ja tietokomponenttien kuvaamisen tarkemmalla tasolla kuin XML-skeemoihin perustuvassa tietomallissa. RDF-tietomallin avulla voidaan määritellä tarkentavia metatietoja asiakirjarakenteiden ja tietokomponenttien ylläpitoa ja hallinnointia varten. XML Schema- ja RDF-pohjaista tiedonmallinnusmenetelmää ei pidä kuitenkaan nähdä toisiaan poissulkevana, koska RDF-tietomallista on mahdollista generoida XML-skeemoja automaattisesti esimerkiksi XSLT-muunnostiedostojen ja SPARQL-kyselyiden avulla.

Suurin etu XHTML+RDFa-standardin soveltamisessa sosiaalihuollon asiakirjojen esittäessä on asiakirjasisältö ja näyttömuoto samassa tiedostossa. Tutkielmassa esitetty menetelmä poistaa tarpeen ylläpitää erillisiä XML-skeemoja ja näyttömuotomäärityksiä. Yksittäinen asiakirja, joka sisältää asiakirjarakenteen sekä asiakirjan esitysmuodon, voidaan nähdä myös todistusvoimaisempana, kun asiakirjan muuttumattomuus voidaan todistaa sähköisen allekirjoituksen avulla.

Yksi jatkotutkimuskohde on sääntöpohjainen validointi toimialakohtaisilla säännöillä, joilla voidaan johtaa uutta tietoa tietosisällöstä. Uutta tietoa voidaan käyttää jatkoprosesseissa, esimerkiksi laskelman tietojen muodostaminen hakemuksessa määritellyistä tuloista ja menoista on mahdollista toimialakohtaisien sääntöjen avulla. Toimialakohtaisten sääntöjen muodostaminen eri käyttötarkoituksiin voisi nopeuttaa ja yhtenäistää sosiaalihuollon prosesseja.

Kyselypohjainen validointi on toinen mahdollinen lähestymistapa XHTML+RDFa-asiakirjojen RDF-tietosisällön validointiin. Kyselyjen muodostaminen SPARQL-kielen avulla on sääntöpohjaista validointia yksinkertaisempaa, mutta kyselyjen määrän kasvaessa validointi voi hidastua merkittävästi sääntöpohjaiseen validointiin verrattuna. Ky-

selypohjaisen validoinnin soveltuvuutta voitaisiin tutkia myös sosiaalihuollon asiakirjasisältöjen validointiin.

Sähköinen tiedonvälitys julkishallinnon toimijoiden välillä on mahdollista, kun päästään sopimukseen yhteisistä tietosisällöistä ja muodostetaan yhteinen tietomalli. Yhteisten sopimusten muodostaminen käytettävästä tietomallista ja viestinvälityksestä on kuitenkin haasteellista, koska yksittäisillä tahoilla ei ole välttämättä vastuuta organisaatiorajoja ylittävästä yhteentoimivuudesta. Tietojärjestelmien yhteentoimivuus on kuitenkin ennen kaikkea poliittinen kysymys. Yhteentoimivat tietojärjestelmät julkishallinnossa edellyttävät valtiotason sitoutumista ja tukea yhteisen tietomallin kehittämiseen, sekä kuntien ja tietojärjestelmätoimittajien velvoittamista yhteisen tietomallin käyttöön.

Nykyiset sosiaalihuollon asiakastietojärjestelmät eivät tue vielä asiakastietojen ja asiakirjojen välittämistä sähköisesti eri tietojärjestelmätoimittajien kehittämien asiakastietojärjestelmien välillä. Sosiaalihuollon tai muiden julkisen hallinnon toimijoiden tietojärjestelmiä ei ole suunniteltu kansallisen yhteentoimivuuden näkökulmasta. Tietojärjestelmien saumaton yhteentoimivuus edellyttää avointa ja yhtenäistä käsitepohjaa, jota ei ole mahdollista toteuttaa tietojärjestelmätoimittajien suljettujen tietomallien pohjalta.

Sosiaalihuollon asiakastietomallin jatkokehittäminen, käyttöönotto ja ylläpito edellyttävät laajaa sisällön asiantuntemusta, sekä yhteistyötä teknisten asiantuntijoiden kanssa. Tietokomponenttien ja asiakirjojen muodostaminen tutkielmassa esitellyn menetelmän avulla mahdollistaa joustavan tavan yhteisen asiakastietomallin käyttöönottoon. Tietomallin ylläpito ja tietojen mallinnus esitetyn menetelmän mukaisesti edellyttää kuitenkin laajaa semanttisten teknologioiden tuntemusta.

Tietoteknisen työn yksinkertaistamiseksi on mahdollista luoda metatietorekisteri, joka muodostaa tutkielmassa esitetyn RDF-tietomallin mukaiset määritykset automaattisesti. Metatietorekisterin avulla sisällön asiantuntijat voisivat suoraan mallintaa teknisesti käyttöönotettavia asiakirjarakenteita. Julkisen hallinnon semanttisen yhteentoimivuuden saavuttamiseksi toimialakohtaisien tietorakenteiden tulisi perustua kansallisesti määriteltyihin ydintietoihin, kuten Henkilö ja Organisaatio. Julkisen hallinnon tietoarkkitehtuurissa on suunnitteilla kansallinen metatietopalvelu [JHT11], jonka tarkoituksena on koordinoida toimialakohtaisten tietomäärityksien yhtenäistämistyötä ja tarjota työkaluja yhtenäisten tietomallien muodostamiseen.

RDF-pohjainen tietomalli on varteenotettava vaihtoehto myös julkisen hallinnon metatietopalvelun tietomalliksi. RDF soveltuu hyvin kuvaamaan eri menetelmillä tuotettuja ja erilaisiin käyttötarkoituksiin määriteltyjä tietoja. Joustavan metatietomallin avulla kehitettyä tietomallia voidaan kehittää ja laajentaa myös tulevaisuudessa ilman suuria järjestelmämuutoksia. RDF-tietomalli mahdollistaa sanastojen, tietorakenteiden ja koodistojen linkittämisen toisiinsa yhtenäisellä metakielellä. Yhtenäiseen metakieleen pohjautuvan metatietopalvelun avulla voidaan toteuttaa tietomääritysten joustavampi ylläpito ja kehitys, kun kaikkia määrityksiä voidaan käsitellä samoilla työkaluilla ja ohjelmistokirjastoilla.

Tikesos-hankkeessa tehty sosiaalihuollon tietosisältöjen yhtenäistämistyö on ensiarvoisen tärkeää sosiaalihuollon tietojärjestelmien yhteentoimivuuden mahdollistamisessa. Sosiaalihuollon yhteinen asiakastietomalli on ensimmäinen askel kohti semanttista yhteentoimivuutta ja kuntarajoja ylittävää asiakastietojen välitystä. Sosiaalihuollon asiakastietomalli sisältää myös koko julkiselle hallinnolle yhteisiä tietoelementtejä, joita voidaan hyödyntää tulevassa julkisen hallinnon tietoarkkitehtuurityössä. Julkisen hallinnon tietoarkkitehtuurin tavoitetilaan on vielä pitkä matka, mutta kehityssuunta on oikea, kunhan muistetaan, että ilman avoimuutta ei voida saavuttaa kansallista yhteentoimivuutta.

LÄHTEET

[AlH08] Allemang D ja Hendler J.: Semantic Web for the Working Ontologists: *Effective Modeling in RDFS and OWL*. Morgan Kaufmann. 2008

[AlK11] Alonen M. ja Konstantin H.: Suitability of Metadata Workbench for data modeling in Finnish social services: *Test report*. Tikesos-hanke. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/cd042e8e-2c96-4b3a-95ee-580c6eca77d1/Suitability+of+Metadata+for+data+modeling+in+Finnish+Social+Services.pdf

[AKS11] Alonen M., Konstantin H. ja Korhonen S.: *XHTML+RDFa-standardin soveltuvuus osaksi sosiaalihuollon asiakirjastandardia*. Tikesos-Hanke. 2.5.2011. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/5d7d320d-6e95-45d8-ae91-0385ebe37700/XHTML-selvitys.pdf

[AKo11] Alonen M. ja Konstantin H.: *Sosiaalihuollon asiakasasiakirjojen näyttömuodot*. Tikesos-hanke. 29.12.2011. Viitattu 29.12.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/620dcda8-8dad-406b-83ae-06f2f6eec387/Sosiaalihuollon+asiakasasiakirjojen+n%c3%a4ytt%c3%b6muodot.pdf

[ATK08] ATK-sanakirja 1. Talentum. 2008.

[AnH04] Antoniou G. ja Harmelen F.: *Semantic Web Primer*. MIT Press, 2004

[Ber02] Berners-Lee T.: Semantic Web: *A new form of the Web that is meaningful to computers will unleash evolution of new possibilities*. Scientific American Special Online Issue. Toukokuu 2002.

[BiH10] Biersteker, H., Hodgson, R.: The Netherlands Ministry of Justice Metadata Workbench: Composing XML Message Schemas from OWL Models. Enterprise Journal Julkaistu 3.5.2010. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.enterprisedatajournal.com/article/netherlands-ministry-justice-metadata-workbench-composing-xml-message-schemas-owl-models.htm

[BLH08] Berrueta, D., Labra, J.E. ja Herman.: XSLT+SPARQL: *Scripting the Semantic Web with SPARQL embedded into XSLT stylesheets*., I. Proceedings of 4th Workshop on Scripting for the Semantic Web. Tenerife, Kesäkuu 2008. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.semanticscripting.org/SFSW2008/papers/1.pdf

[CCT09] *Core Components Technical Spesification*, Version: 3.0, 29.11.2009, Viitattu 8.11.2011, saatavilla osoitteesta: http://www.unece.org/cefact/codesfortrade/CCTS/CCTS-Version3.pdf

[CEF09] *XML Naming and Desing Rules Technical Spesification*, Version 3.0,. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.unece.org/cefact/xml/UNCEFACT+XML+NDR+V3p0.pdf

[DBP11] DBPedia:. RDF-muotoinen tietokanta wikipediasta. Viitattu 8.11.2011, saatavilla osoitteesta: http://dbpedia.org

[DOM11] W3C.: Document Object Model. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/DOM/

[EYE11] Eyeball: checking RDF/OWL for common problems. Viitattu 8.11.2011, saatavilla osoitteesta: http://jena.sourceforge.net/Eyeball/

[GrM05] Grimm S. ja Motik B.: *ClosedWorld Reasoning in the SemanticWeb through Epistemic Operators*. In Bernardo Cuenca Grau, Ian Horrocks, Bijan Parsia, and Peter Patel-Schneider, editors, Proc. of the Workshop on OWL: Experiences and Directions (OWLED 2005), Galway, Ireland. 2005.

[HAK11] Hypponen K., Alonen M., Korhonen S. ja Hotti V.: *XHTML with RDFa as a Semantic Document Format for CCTS Modelled Documents and its Application for Social Services*. ESWC 2011 Workshops, LNCS 7117 Proceedings, 2011.

[HHL+09] Huttunen R., Hyppönen K., Lehmuskoski A., Hotti V., Nevalainen J., Tossavainen P., Ailio E. ja Miettinen A.: *Sosiaalihuollon asiakastietojen ja -asiakirjojen tietomallinnus.* 21.5.2009. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/fd06b2f6-8a2a-4e83-9791-94de48b3260d/Sosiaalihuollon+asiakastietojen+ja+-asiakirjojen+tietomallinnus.pdf

[HHM+09] Huttunen R., Hyppönen K., Martikainen I., Hotti V. ja Nevalainen J.: *Sosiaalihuollon asiakasasiakirjojen tekniset määritykset.* 9.6.2009. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/58555ac4-c585-44c2-aba5-4168db8af01e/Sosiaalihuollon+Asiakasasiakirjojen+Tekniset+M%c3%a4%c3%a4ritykset.pdf

[HHK+08] Hotti V., Huttunen R., Kajander A., Lehmuskoski A., Ojala M., Taskinen T., Tiihonen T.: *Tietämyksenhallinta ja ontologiat sosiaalihuollon näkökulmasta*. Sosiaali- ja terveysalan tietotyhteiskuntayksikön julkaisuja. 2008.

[HHL+10] Huttunen R., Hyppönen K., Lehmuskoski A., Hotti V., Nevalainen J., Ailio E., Miettinen A.: *Sosiaalihuollon asiakastietojen ja -asiakirjojen tietomallinnus*. 2010. Viitattu 8.11.2011, saatavilla osoitteessa: http://www.sosiaaliportti.fi/File/fd06b2f6-8a2a-4e83-9791-94de48b3260d/Sosiaalihuollon+asiakastietojen+ja+-asiakirjojen+tietomallinnus.pdf

[HPP+05] Horrocks I., Parsia B., Patel-Schneider P., Hendler J.: Semantic Web Architecture: *Stack or Two Towers?* Principles and Practice of Semantic Web Reasoning. Lecture Notes in Computer Science, 2005, Volume 3703/2005, 37-41.

[HLT+11] Ailio E., Häkälä N., Laaksonen P., Tossavainen P., Suhonen M., Suhonen S.: *Sosiaalihuollon asiakasasiakirjasanasto.* Heinäkuu 2011. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/8cbe843b-cbd8-4b85-8e58-ee584c624aa6/Sosiaalihuollon+asiakasasiakirjasanasto.pdf

[HLL+08] Hämäläinen P., Lehto H., Lehtonen J., Ojala M., Palojoki S.: *Koodistopalvelun käsikirja*. 2007. Viitattu 8.11.2011, saatavilla osoitteesta: http://sty.stakes.fi/NR/rdonlyres/C0432B2A-7AB6-4D4E-AFE2-3927962AE423/16513/KopioKoodistopalveluk%C3%A4sikirja02_hanna_pdf_aikio.pdf

[HTM99] W3C.: HTML 4.01 Specification. W3C Recommendation 24 December 1999. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/REC-html40/

[ISO10] ISO/IEC 11179 Metadata Registries. Viitattu 8.11.2011, saatavilla osoitteesta: http://metadata-stds.org/11179/

[JEN11] Jena 2 Inference support. Viitattu 8.11.2011, saatavilla osoitteesta: http://jena.sourceforge.net/inference/index.html

[JHT11] Julkisen hallinnon tietoarkkitehtuuri. Valtiovarainministeriö. Julkaistu 12.11.2010. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/03_muut_asiakirjat/20100105Valtio/05_Julkishallinnon_tietoarkkitehtuuri.pdf

[JHS10] JHS 175: Julkisen hallinnon sanastotyöprosessi, Versio: 1.0, 23.04.2010. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.jhs-suositukset.fi/web/guest/jhs/recommendations/175

[JHS09] JHS 170: Julkishallinnon XML-skeemat, Versio: 1.0, 30.03.2009. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.jhs-suositukset.fi/web/guest/jhs/recommendations/170

[JQR11] JQuery javascript library. Viitattu 8.11.2011, saatavilla osoitteesta: http://jquery.com

[KBB+05] Kifer, M.; de Bruijn, J.; Boley, Harold; Fensel, D.: A Realistic Architecture for the Semantic Web. Proceedings of the International Conference on Rules and Rule Markup Languages for the Semantic Web. RuleML. 2005.

[LAU10] Laudi, A.: The semantic interoperability centre europe: *Reuse and the negotiation of meaning*. In: Charalabidis, Y. Interoperability in Digital Pub-

lic Services and Administration: Bridging E-Government and E-Business, pp. 144{161. IGIGlobal. 2010

[Lyb06] Lybeck J. et al.: Arkistot yhteiskunnan toimiva muisti: *Asiakirjahallinnon ja arkistotoimen oppikirja*. Arkistolaitoksen toimituksia 2006. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.arkisto.fi/uploads/Palvelut/Julkaisut/asiakirjahallinnon_oppikirja.pdf

[LHT+08] Lehmuskoski A., Huovila M., Tiihonen T., Taskinen T., Savolainen J., Hotti V., Paakkanen E., Mykkänen J.: *Vaihtoehdot sosiaalihuollon asiakasasiakirjojen tekniseksi standardiksi.* Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/0a4fcb9c-b5ee-4c53-b9f9-3029a4933455/Sosiaalihuollon+teknisen+standardin+implementointisuunnitelma.pdf

[LKP+11] Laaksonen M., Kääriäinen Aino., Penttilä M., Tapola-Haapala M., Sahala H., Kärki J., Jäppinen A.: *Asiakastyön dokumentointi sosiaalihuollossa*. Tikesos-hanke. 2011. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/fbf710a0-e1cd-4799-a05e-3913e8754992/Asiakasty%c3%b6n+dokumentointi+sosiaalihuollossa.pdf

[LeK07] Lehomuskoski A., Kuusisto-Niemi A.: *Sosiaalialan sanasto asiakastietojärjestelmää varten*. Huhtikuu 2007. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/9150c01f-cce2-4a56-b4ee-d529e77d3660/Sanasto.pdf

[LPS10] Liegl P., Pichler C., Strommer M.: *State-of-the-art in business document standards*. 2010 IEEE

[MHS09] Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. Web Semantics: Science, Services and Agents on the World Wide Web 7(2), 74-89 (2009)

[NIE07] NIEM introduction. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.niem.gov/files/NIEM_Introduction.pdf/

[OWL09] W3C.: OWL 2. Web Ontology Language Primer, W3C Recommendation 27 October 2009 Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/owl2-primer/

[Paa07] Paakkanen E. et al. *Sosiaalialan tieojärjestelmästandardien kartoitus.* 2.2.2007. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/1c75ef99-423a-483d-8d24-1393ddc51ac1/Standardikartoitus.pdf

[RDC10] W3C.: RDFa Core 1.1: *Syntax and processing rules for embedding RDF trough attributes*. W3C Working Draft 22 April 2010

[RDA10] W3C.: *RDFa in XHTML: Syntax and Processing.* W3C Recommendation 14 October

[RDC11] RDFa Core 1.1 implementation in Java. Viitattu 8.11.2011, saatavilla osoitteesta: http://code.google.com/p/rdfa-core-java/

[RDS04] W3C.: RDF Vocabulary Description Language 1.0: *RDF Schema*. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/rdf-schema/

[RDF04] W3C.: *RDF Primer*, W3C Recommendation 10 February 2004. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/rdf-syntax/

[Rei78] Reiter, R.: *On closed world data bases*. In: Gaillaire, H., Minker, J. (eds.) Logicand Data Bases, pp. 55{76. Plenum Press, New York. 1978.

[RIF10] W3C.: *RIF Overview*. W3C Working Group Note 22 June 2010. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/rif-overview/

[STM05] *Sosiaali- ja terveysministeriön monisteita* 2005:1. ISBN 952-00-1619-8

[SAX11] Simple API for XML. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.saxproject.org/

[SiT09] Sirin, E., Tao, J.: *Towards integrity constraints in OWL*. In: Proceedings of the Workshop on OWL: Experiences and Directions, OWLED 2009 (2009)

[STS11] Sosiaalihuollon tietokomponenttien sanasto. 2011.

[SPL11] Sosiaalipalvelujen luokituksen sanasto. Versio 1.0. Julkaistu 12.10.2011. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.sosiaaliportti.fi/File/07a7a8b2-f4c6-4e28-a5f9-d84d5289fa81/Sosiaalipalvelujen+luokituksen+sanasto.pdf

[Sow99] Sowa J., *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA, ©2000. 1999.

[SRQ08] W3C.: *SPARQL Query Language for RDF*. W3C Recommendation 15 January 2008. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/rdf-sparql-query/

[SRX08] W3C.: *SPARQL Query Results XML Format*. W3C Recommendation 15 January 2008. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/rdf-sparql-XMLres/

[SGM86] ISO 8879:1986. *Standard Generalized Markup Language*. International Organization for Standardization. 1986.

[SCH06] ISO/IEC 19757-3:2006 Information technology -- *Document Schema Definition Language* (DSDL) -- Part 3: Rule-based validation - Schematron. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.schematron.com/

[TEP11] TEPA – Sanastokeskus TSK:n termipankki. Viitattu 8.11.2011, saatavilla osoitteesta: www.tsk.fi/tepa

[UNC07] *Unicode in XML and other Markup Languages*. W3C Working Group. 2007. Viitattu 8.11.2011, saatavilla osoitteesta: http://unicode.org/reports/tr20/

[VAL11] Validointipalvelu. Viitattu 8.11.2011, saatavilla osoitteesta: http://code.google.com/p/rdf-rule-validator/

[YHT11] Yhteentoimivuus.fi Sanasto. Viitattu 8.11.2011, saatavilla osoitteesta: https://www.yhteentoimivuus.fi/view/smeta/Sanasto.xhtml

[XHO11] XMLHttpRequest. W3C Candidate Recommendation 3 August 2010. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/XMLHttpRequest/

[XFC11] XML for CCTS. Viitattu 8.11.2011, saatavilla osoitteesta: http://www1.unece.org/cefact/platform/display/ATG/ATG2+XML+for+CCTS

[XHT02] W3C.: XHTML™ 1.0 *The Extensible HyperText Markup Language*, W3C Recommendation 26 January 2000, revised 1 August 2002. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/xhtml11/

[XHM01] W3C.: *XHTML Modularization* - an Overview. 02 February 2000, revised on April 2001. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/MarkUp/modularization

[XML08] W3C.: *Extensible Markup Language (XML) 1.0 (Fifth Edition)*, W3C Recommendation. 2008. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/xml/

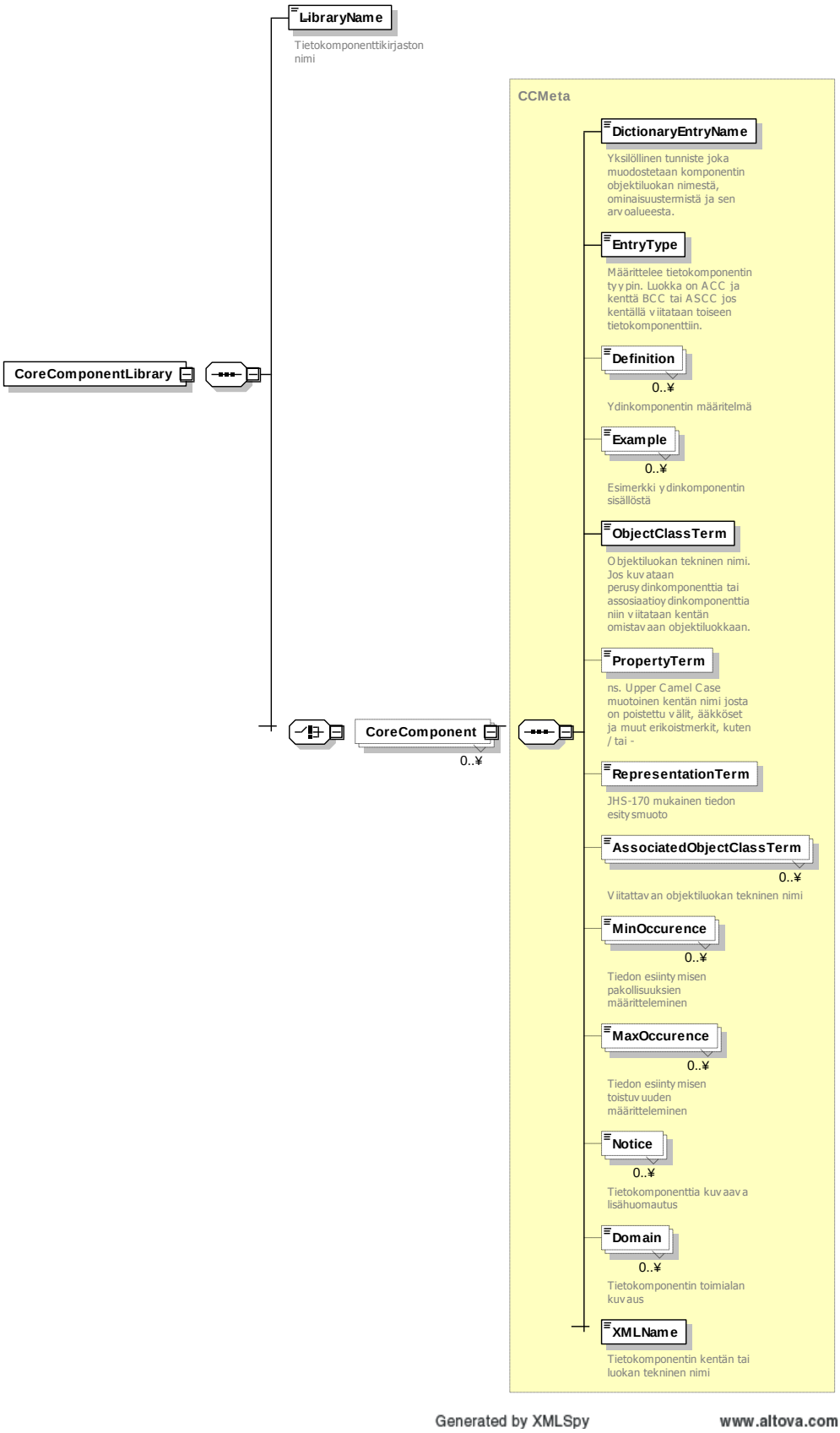
[XPA99] W3C.: *XML Path Language (XPath)* Version 1.0. 1999. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/xpath/

[XQR10] XML Query Language. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/xquery/

[XSD04] W3C.: *XML Schema Part 0: Primer Second Edition*. W3C Recommendation. 2004. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/xmlschema-0/

[XSL07] W3C.: *XSL Transformations* (XSLT) Version 2.0. W3C Recommendation. 2007. Viitattu 8.11.2011, saatavilla osoitteesta: http://www.w3.org/TR/xslt20

A Tietokomponenttikirjaston CCTS XML -skeema



B Esimerkki tietokomponentista CCTS XML -muodosta

```
<?xml version="1.0" encoding="UTF-8"?>
<CCTSLibrary>
  <LibraryName>Tietokomponentit</LibraryName>
  <CoreComponent>
    <DictionaryEntryName>Aikavali</DictionaryEntryName>
    <Definition>tiettyjen vuorokauteen sisältyvien kellonaikojen välinen aika</Definition>
    <Notice>Vuorokausi tarkoittaa tässä 24 tunnin mittaista aikaa. Aikaväli voi olla esimerkiksi
klo 8-14.</Notice>
    <Label>Aikaväli</Label>
    <EntryType>ACC</EntryType>
    <ObjectClassTerm>Aikavali</ObjectClassTerm>
  </CoreComponent>
  <CoreComponent>
    <DictionaryEntryName>Aikavali.Alkamisaika.Aika</DictionaryEntryName>
    <Label>Alkamisaika</Label>
    <EntryType>BCC</EntryType>
    <Definition>aikavälin alkava kellonaika</Definition>
    <ObjectClassTerm>Aikavali</ObjectClassTerm>
    <PropertyTerm>Alkamisaika</PropertyTerm>
    <RepresentationTerm>Aika</RepresentationTerm>
    <MinOccurence>0</MinOccurence>
    <MaxOccurence>1</MaxOccurence>
    <Reference>Siirretty ydinkomponenteista 10.11.2009</Reference>
    <Example>8.15</Example>
    <Notice>Alkutunti voi olla esimerkiksi 8, 8.15 tai 8.30.</Notice>
    <XMLName>AlkamisAika</XMLName>
  </CoreComponent>
  <CoreComponent>
    <DictionaryEntryName>Aikavali.Paattymisaika.Aika</DictionaryEntryName>
    <Label>Päättymisaika</Label>
    <EntryType>BCC</EntryType>
    <Definition>aikavälin päättävä kellonaika</Definition>
    <ObjectClassTerm>Aikavali</ObjectClassTerm>
    <PropertyTerm>Paattymisaika</PropertyTerm>
    <RepresentationTerm>Aika</RepresentationTerm>
    <MinOccurence>0</MinOccurence>
    <MaxOccurence>1</MaxOccurence>
    <Reference>Siirretty ydinkomponenteista 10.11.2009</Reference>
    <Example>14.30</Example>
    <Notice>Lopputunti voi olla esimerkiksi 14, 14.15 tai 14.30.</Notice>
    <XMLName>PaattymisAika</XMLName>
  </CoreComponent>
  <CoreComponent>
    <DictionaryEntryName>Aikavali.Alkamispaiva.Pvm</DictionaryEntryName>
    <Label>Alkamispäivä</Label>
    <EntryType>BCC</EntryType>
    <Definition>ajanjakson aloittava päivämäärä</Definition>
    <ObjectClassTerm>Aikavali</ObjectClassTerm>
    <PropertyTerm>Alkamispaiva</PropertyTerm>
    <RepresentationTerm>Pvm</RepresentationTerm>
    <MinOccurence>0</MinOccurence>
    <MaxOccurence>1</MaxOccurence>
    <Reference>Ajanvaraus-tietokomponentti</Reference>
    <Example>21.4.2010</Example>
    <Notice>Jos aikaväli alkaa 21.4.2010 klo 21 ja päättyy 22.4.2010 klo 03, tämän kentän arvo
on 21.4.2010.</Notice>
    <XMLName>AlkamispaivaPvm</XMLName>
  </CoreComponent>
</CCTSLibrary>
```

C CCTS RDF tietomallin XSLT-muunnostiedosto

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fn="http://www.w3.org/2005/xpath-functions" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
  <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
  <xsl:template match="text()"/>
  <xsl:template match="CCTSLibrary">
    <xsl:variable name="LibraryName" select="LibraryName"/>
    <xsl:variable name="StaticNamespace" select="'http://sosmeta.fi/ontologies/'"/>
    <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:ccts="http://tikesos.fi/ccts#"
xmlns:sos="http://sosmeta.fi/ontologies/sos#" xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base="http://sosmeta.fi/ontologies/sos#">
      <rdf:Description rdf:about="#{replace($LibraryName,' ',' ')}">
        <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
        <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
          <xsl:value-of select="$LibraryName"/>
        </rdfs:label>
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
      </rdf:Description>
      <rdf:Description rdf:about="#ACC">
        <rdfs:subClassOf rdf:resource="#{replace($LibraryName,' ','_')}"/>
        <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
          <xsl:value-of select="$LibraryName"/>
        </rdfs:label>
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
      </rdf:Description>
      <rdf:Description rdf:about="#ASCC">
        <rdfs:subClassOf rdf:resource="#{replace($LibraryName,' ','_')}"/>
        <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
          <xsl:value-of select="$LibraryName"/>
        </rdfs:label>
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
      </rdf:Description>
      <rdf:Description rdf:about="#BCC">
        <rdfs:subClassOf rdf:resource="#{replace($LibraryName,' ',' ')}"/>
        <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
          <xsl:value-of select="$LibraryName"/>
        </rdfs:label>
        <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
      </rdf:Description>
      <xsl:for-each-group select="CoreComponent" group-adjacent="ObjectClassTerm">
        <xsl:variable name="groupKey" select="current-grouping-key()"/>
        <xsl:variable name="ObjectClassTerm" select="$groupKey"/>
        <xsl:for-each select="current-group()">
          <xsl:variable name="EntryType" select="EntryType"/>
          <xsl:variable name="XMLName" select="XMLName"/>
          <xsl:variable name="PropertyXMLName"
              select="concat(lower-case(substring($XMLName, 1, 1)), substring($XMLName, 2))"/>
          <xsl:variable name="PropertyTerm" select="PropertyTerm"/>
          <xsl:variable name="MinOccurence" select="MinOccurence"/>
          <xsl:variable name="MaxOccurence" select="MaxOccurence"/>
          <xsl:variable name="Label" select="Label"/>
          <xsl:variable name="Definition" select="Definition"/>
          <xsl:variable name="Notice" select="Notice"/>
          <xsl:variable name="DictionaryEntryName">
            <xsl:value-of select="DictionaryEntryName"/>
          </xsl:variable>
          <xsl:variable name="Example" select="Example"/>
          <xsl:choose>
            <xsl:when test="$EntryType='ACC'">
              <rdf:Description rdf:about="#{current-grouping-key()}">
                <xsl:for-each select="current-group()">
                  <xsl:variable name="EntryType" select="EntryType"/>
                  <xsl:variable name="XMLName" select="XMLName"/>
                  <xsl:variable name="Definition" select="Definition"/>
                  <xsl:variable name="Example" select="Example"/>
                  <xsl:variable name="Group" select="Group"/>
                  <xsl:variable name="Domain" select="Domain"/>
                  <xsl:variable name="Notice" select="Notice"/>
                  <xsl:variable name="Label" select="Label"/>
                  <xsl:choose>
                    <xsl:when test="$EntryType='ACC'">
```

```
                    <rdfs:subClassOf rdf:resource="#ACC"/>
                    <rdf:type rdf:resource="http://tikesos.fi/ccts#{$EntryType}"/>
                    <ccts:hasNext rdf:resource="#{following-
sibling::CoreComponent[1]/DictionaryEntryName}"/>
                    <xsl:if test="$Definition">
                      <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$Definition"/>
                      </ccts:definition>
                    </xsl:if>
                    <xsl:if test="$Example">
                      <ccts:example rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$Example"/></ccts:example>
                    </xsl:if>
                    <xsl:if test="$Group">
                      <ccts:group rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$Group"/>ccts:group>
                    </xsl:if>
                    <xsl:if test="$Domain">
                      <ccts:domain rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$Domain"/>
                      </ccts:domain>
                    </xsl:if>
                    <xsl:if test="$Notice">
                      <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$Notice"/>
                      </ccts:notice>
                    </xsl:if>
                    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                      <xsl:value-of select="$Label"/></rdfs:label>
                    <ccts:hasDictionaryEntryName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                      <xsl:value-of select="$DictionaryEntryName"/>
                    </ccts:hasDictionaryEntryName>
                    <ccts:hasObjectClassTerm rdf:resource="#{$groupKey}"/>
                  </xsl:when>
                  <xsl:otherwise>
                    <rdfs:subClassOf rdf:nodeID="ID{concat($groupKey,$XMLName)}"/>
                  </xsl:otherwise>
                </xsl:choose>
              </xsl:for-each>
            </rdf:Description>
          </xsl:when>
          <xsl:otherwise>
            <rdf:Description rdf:about="#{$DictionaryEntryName}">
              <rdf:type rdf:resource="http://tikesos.fi/ccts#{$EntryType}"/>
              <xsl:if test="following-sibling::CoreComponent[1]/ObjectClassTerm=$groupKey">
                <ccts:hasNext rdf:resource="#{following-sibling::CoreComponent[1]/DictionaryEntryName}"/>
              </xsl:if>
              <xsl:if test="$Definition">
                <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$Definition"/>
                </ccts:definition>
              </xsl:if>
              <xsl:if test="$Example">
                <ccts:example rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$Example"/>
                </ccts:example>
              </xsl:if>
              <xsl:if test="$Notice">
                <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$Notice"/>
                </ccts:notice>
              </xsl:if>
              <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                <xsl:value-of select="$Label"/>
              </rdfs:label>
              <ccts:hasDictionaryEntryName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                <xsl:value-of select="$DictionaryEntryName"/>
              </ccts:hasDictionaryEntryName>
              <ccts:hasObjectClassTerm rdf:resource="#{$groupKey}"/>
              <ccts:hasPropertyTerm rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                <xsl:value-of select="$PropertyTerm"/>
              </ccts:hasPropertyTerm>
              <ccts:minOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
                <xsl:value-of select="$MinOccurence"/>
              </ccts:minOccurs>
              <ccts:maxOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                <xsl:value-of select="$MaxOccurence"/>
```

```
            </ccts:maxOccurs>
          <xsl:choose>
              <xsl:when test="EntryType='BCC'">
                <rdfs:subClassOf rdf:resource="#BCC"/>
                <xsl:variable name="RepresentationTerm" select="RepresentationTerm"/>
                <ccts:onProperty rdf:resource="#{$PropertyXMLName}"/>
                <ccts:hasRepresentationTerm rdf:resource="http://jhs-
suositukset.fi#{$RepresentationTerm}"/>
              </xsl:when>
              <xsl:when test="$EntryType='ASCC'">
                <rdfs:subClassOf rdf:resource="#ASCC"/>
                <xsl:variable name="AssociatedObjectClassTerm" select="AssociatedObjectClassTerm"/>
                <xsl:variable name="AssociatedObjectClassTerm" select="$AssociatedObjectClassTerm"/>
                <ccts:onProperty rdf:resource="#{$PropertyXMLName}"/>
                <xsl:variable name="Namespace" select="Namespace"/>
                <ccts:hasAssociatedObjectClass rdf:resource="{concat('#',$AssociatedObjectClassTerm)}"/>
              </xsl:when>
            </xsl:choose>
          </rdf:Description>
        </xsl:otherwise>
      </xsl:choose>
      <xsl:choose>
        <xsl:when test="$EntryType='ASCC'">
          <xsl:variable name="AssociatedObjectClassTerm" select="AssociatedObjectClassTerm"/>
          <rdf:Description rdf:nodeID="ID{concat($groupKey,$XMLName)}">
            <xsl:choose>
              <xsl:when test="$MinOccurence=1">
                <xsl:choose>
                  <xsl:when test="$MaxOccurence=1">
                    <owl:qualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:qualifiedCardinality>
                  </xsl:when>
                  <xsl:otherwise>
                    <owl:minQualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minQualifiedCardinality>
                  </xsl:otherwise>
                </xsl:choose>
              </xsl:when>
              <xsl:otherwise>
                <xsl:choose>
                  <xsl:when test="$MaxOccurence=1">
                    <owl:maxQualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxQualifiedCardinality>
                  </xsl:when>
                  <xsl:otherwise>
                    <owl:minQualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minQualifiedCardinality>
                  </xsl:otherwise>
                </xsl:choose>
              </xsl:otherwise>
            </xsl:choose>
            <owl:onProperty rdf:resource="#{$PropertyXMLName}"/>
            <xsl:variable name="Namespace" select="Namespace"/>
            <owl:onClass rdf:resource="{concat('#',$AssociatedObjectClassTerm)}"/>
            <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
          </rdf:Description>
          <xsl:if test="not(preceding-sibling::CoreComponent[XMLName=$XMLName])">
            <rdf:Description rdf:about="#{$PropertyXMLName}">
              <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                <xsl:value-of select="$Label"/>
              </rdfs:label>
              <rdf:type rdf:resource="http://tikesos.fi/ccts#ASCCProperty"/>
            </rdf:Description>
          </xsl:if>
        </xsl:when>
        <xsl:when test="$EntryType='BCC'">
          <rdf:Description rdf:nodeID="ID{concat($groupKey,$XMLName)}">
            <xsl:choose>
              <xsl:when test="$MinOccurence=1">
                <xsl:choose>
                  <xsl:when test="$MaxOccurence=1">
                    <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:cardinality>
                  </xsl:when>
                  <xsl:otherwise>
                    <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minCardinality>
```

```
                    </xsl:otherwise>
                  </xsl:choose>
                </xsl:when>
              <xsl:otherwise>
                  <xsl:choose>
                    <xsl:when test="$MaxOccurence=1">
                      <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxCardinality>
                    </xsl:when>
                    <xsl:otherwise>
                      <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
                    </xsl:otherwise>
                  </xsl:choose>
                </xsl:otherwise>
              </xsl:choose>
              <owl:onProperty rdf:resource="#{$PropertyXMLName}"/>
              <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
            </rdf:Description>
            <xsl:if test="not(preceding-sibling::CoreComponent[XMLName=$XMLName])">
              <xsl:variable name="RepresentationTerm" select="RepresentationTerm"/>
              <rdf:Description rdf:about="#{$PropertyXMLName}">
                <xsl:choose>
                  <xsl:when test="$RepresentationTerm='Teksti'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Numero'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Nimi'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Pvm'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Hetki'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Kytkin'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Lkm'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Maara'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Aika'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#time"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Prosentti'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Arvo'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                  </xsl:when>
                  <xsl:when test="$RepresentationTerm='Koodi'">
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                  </xsl:when>
                  <xsl:otherwise>
                    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                  </xsl:otherwise>
                </xsl:choose>
                <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$Label"/>
                </rdfs:label>
                <rdf:type rdf:resource="http://tikesos.fi/ccts#BCCProperty"/>
              </rdf:Description>
            </xsl:if>
          </xsl:when>
        </xsl:choose>
      </xsl:for-each>
    </xsl:for-each-group>
  </rdf:RDF>
 </xsl:template>
</xsl:stylesheet>
```

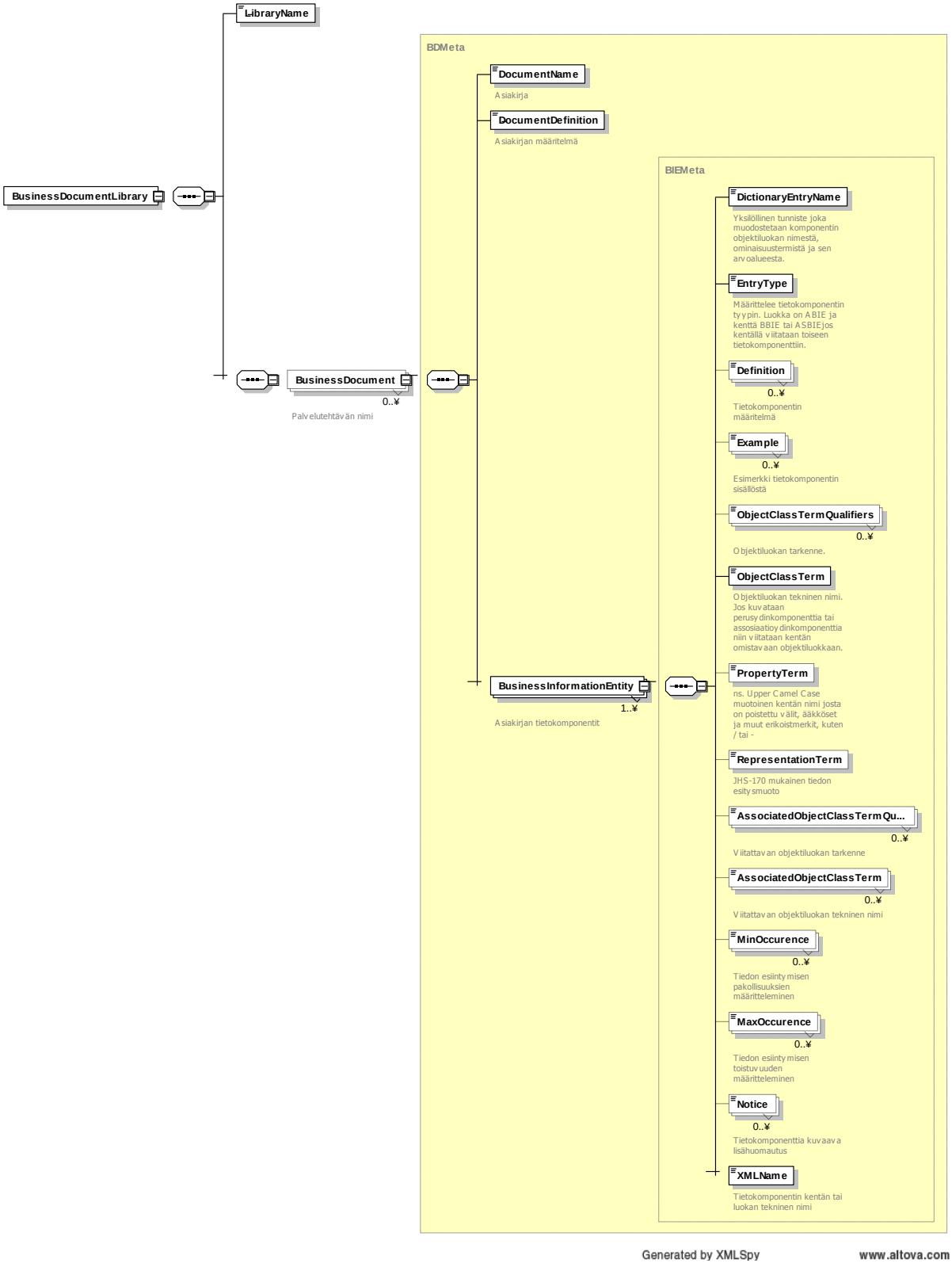
D Esimerkki CCTS-ontologian mukaisesta RDF-tietomallista

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:ccts="http://tikesos.fi/ccts#" xmlns:fn="http://www.w3.org/2005/xpath-functions"
xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:sos="http://sosmeta.fi/ontologies/sos#" xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xml:base="http://sosmeta.fi/ontologies/sos#">
  <rdf:Description rdf:about="#Tietokomponentit">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Tietokomponentit</rdfs:label>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="#ACC">
    <rdfs:subClassOf rdf:resource="#Tietokomponentit"/>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Tietokomponentit</rdfs:label>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="#ASCC">
    <rdfs:subClassOf rdf:resource="#Tietokomponentit"/>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Tietokomponentit</rdfs:label>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="#BCC">
    <rdfs:subClassOf rdf:resource="#Tietokomponentit"/>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Tietokomponentit</rdfs:label>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="#Aikavali">
    <rdfs:subClassOf rdf:resource="#ACC"/>
    <rdf:type rdf:resource="http://tikesos.fi/ccts#ACC"/>
    <ccts:hasNext rdf:resource="#Aikavali.Alkamisaika.Aika"/>
    <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">tiettyjen vuorokau-
teen sisältyvien kellonaikojen välinen aika</ccts:definition>
    <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Vuorokausi tarkoittaa
tässä 24 tunnin mittaista aikaa. Aikaväli voi olla esimerkiksi klo 8-14.</ccts:notice>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Aikaväli</rdfs:label>
    <ccts:hasDictionaryEntryName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Aikavali</ccts:hasDictionaryEntryName>
    <ccts:hasObjectClassTerm rdf:resource="#Aikavali"/>
    <rdfs:subClassOf rdf:nodeID="IDAikavaliAlkamisAika"/>
    <rdfs:subClassOf rdf:nodeID="IDAikavaliPaattymisAika"/>
    <rdfs:subClassOf rdf:nodeID="IDAikavaliAlkamispaivaPvm"/>
  </rdf:Description>
  <rdf:Description rdf:about="#Aikavali.Alkamisaika.Aika">
    <rdf:type rdf:resource="http://tikesos.fi/ccts#BCC"/>
    <ccts:hasNext rdf:resource="#Aikavali.Paattymisaika.Aika"/>
    <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">aikavälin alkava
kellonaika</ccts:definition>
    <ccts:example rdf:datatype="http://www.w3.org/2001/XMLSchema#string">8.15</ccts:example>
    <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkutunti voi olla esi-
merkiksi 8, 8.15 tai 8.30.</ccts:notice>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkamisaika</rdfs:label>
    <ccts:hasDictionaryEntryName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Aikavali.Alkamisaika.Aika</ccts:hasDictio
naryEntryName>
    <ccts:hasObjectClassTerm rdf:resource="#Aikavali"/>
    <ccts:hasPropertyTerm
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkamisaika</ccts:hasPropertyTerm>
```

```
    <ccts:minOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</ccts:minOccurs>
    <ccts:maxOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#string">1</ccts:maxOccurs>
    <rdfs:subClassOf rdf:resource="#BCC"/>
    <ccts:onProperty rdf:resource="#alkamisAika"/>
    <ccts:hasRepresentationTerm rdf:resource="http://jhs-suositukset.fi#Aika"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="IDAikavaliAlkamisAika">
    <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxCardinality>
    <owl:onProperty rdf:resource="#alkamisAika"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
  </rdf:Description>
  <rdf:Description rdf:about="#alkamisAika">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#time"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkamisaika</rdfs:label>
    <rdf:type rdf:resource="http://tikesos.fi/ccts#BCCProperty"/>
  </rdf:Description>
  <rdf:Description rdf:about="#Aikavali.Paattymisaika.Aika">
    <rdf:type rdf:resource="http://tikesos.fi/ccts#BCC"/>
    <ccts:hasNext rdf:resource="#Aikavali.Alkamispaiva.Pvm"/>
    <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">aikavälin päättävä
kellonaika</ccts:definition>
    <ccts:example rdf:datatype="http://www.w3.org/2001/XMLSchema#string">14.30</ccts:example>
    <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Lopputunti voi olla esi-
merkiksi 14, 14.15 tai 14.30.</ccts:notice>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Päättymisaika</rdfs:label>
    <ccts:hasDictionaryEntryName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Aikavali.Paattymisaika.Aika</ccts:hasDict
ionaryEntryName>
    <ccts:hasObjectClassTerm rdf:resource="#Aikavali"/>
    <ccts:hasPropertyTerm
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Paattymisaika</ccts:hasPropertyTerm>
    <ccts:minOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</ccts:minOccurs>
    <ccts:maxOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#string">1</ccts:maxOccurs>
    <rdfs:subClassOf rdf:resource="#BCC"/>
    <ccts:onProperty rdf:resource="#paattymisAika"/>
    <ccts:hasRepresentationTerm rdf:resource="http://jhs-suositukset.fi#Aika"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="IDAikavaliPaattymisAika">
    <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxCardinality>
    <owl:onProperty rdf:resource="#paattymisAika"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
  </rdf:Description>
  <rdf:Description rdf:about="#paattymisAika">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#time"/>
    <rdfs:label
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Päättymisaika</rdfs:label>
    <rdf:type rdf:resource="http://tikesos.fi/ccts#BCCProperty"/>
  </rdf:Description>
  <rdf:Description rdf:about="#Aikavali.Alkamispaiva.Pvm">
    <rdf:type rdf:resource="http://tikesos.fi/ccts#BCC"/>
    <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">ajanjakson aloittava
päivämäärä</ccts:definition>
    <ccts:example
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">21.4.2010</ccts:example>
    <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Jos aikaväli alkaa
21.4.2010 klo 21 ja päättyy 22.4.2010 klo 03, tämän kentän arvo on 21.4.2010.</ccts:notice>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkamispäivä</rdfs:label>
    <ccts:hasDictionaryEntryName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Aikavali.Alkamispaiva.Pvm</ccts:hasDictio
naryEntryName>
    <ccts:hasObjectClassTerm rdf:resource="#Aikavali"/>
    <ccts:hasPropertyTerm
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkamispaiva</ccts:hasPropertyTerm>
    <ccts:minOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</ccts:minOccurs>
    <ccts:maxOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#string">1</ccts:maxOccurs>
    <rdfs:subClassOf rdf:resource="#BCC"/>
```

```
    <ccts:onProperty rdf:resource="#alkamispaivaPvm"/>
    <ccts:hasRepresentationTerm rdf:resource="http://jhs-suositukset.fi#Pvm"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="IDAikavaliAlkamispaivaPvm">
    <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxCardinality>
    <owl:onProperty rdf:resource="#alkamispaivaPvm"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
  </rdf:Description>
  <rdf:Description rdf:about="#alkamispaivaPvm">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Alkamispäivä</rdfs:label>
    <rdf:type rdf:resource="http://tikesos.fi/ccts#BCCProperty"/>
  </rdf:Description>
</rdf:RDF>
```

E Asiakirjarakennekirjaston CCTS XML -skeema



F Esimerkki asiakirjarakennekirjastosta CCTS XML -muodossa

```
<?xml version="1.0" encoding="UTF-8"?>
<BusinessDocumentLibrary xmlns="http://www.sosmeta.fi/cctsxml/DocumentLibrary"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.sosmeta.fi/cctsxml/DocumentLibrary DocumentLibrary.xsd">
  <LibraryName>Toimeentulotuki</LibraryName>
  <BusinessDocument>
    <DocumentName>Toimeentulotukihakemus</DocumentName>
    <DocumentDefinition>Asiakirjalla asiakas hakee toimeentulotukea ensimmäistä kertaa tai toistamiseen.
Jättäessään toistamiseen toimeentulotukihakemuksen, asiakas ilmoittaa mahdolliset muutokset olosuhteissaan ja
tarvittaessa korjaa esitäytetyt, edellisen hakemuksen tiedot.</DocumentDefinition>
    <BusinessInformationEntity>
      <DictionaryEntryName>Asiakas.Yksityishenkilo</DictionaryEntryName>
      <EntryType>ASBIE</EntryType>
      <Definition>Toimeentulotuen hakija</Definition>
      <ObjectClassTerm>Toimeentulotukihakemus</ObjectClassTerm>
      <PropertyTerm>AsiakasYksitysihenkilo</PropertyTerm>
      <AssociatedObjectClassTermQualifiers>Asiakas</AssociatedObjectClassTermQualifiers>
      <AssociatedObjectClassTerm>Yksityishenkilo</AssociatedObjectClassTerm>
      <XMLName>AsiakasYksityishenkilo</XMLName>
    </BusinessInformationEntity>
        <BusinessInformationEntity>
      <DictionaryEntryName>Laatija.Yksityishenkilo</DictionaryEntryName>
      <EntryType>ASBIE</EntryType>
      <ObjectClassTerm>Toimeentulotukihakemus</ObjectClassTerm>
      <PropertyTerm>LaatijaYksitysihenkilo</PropertyTerm>
      <AssociatedObjectClassTermQualifiers>Laatija</AssociatedObjectClassTermQualifiers>
      <AssociatedObjectClassTerm>Yksityishenkilo</AssociatedObjectClassTerm>
      <XMLName>LaatijaYksityishenkilo</XMLName>
    </BusinessInformationEntity>
    <BusinessInformationEntity>
      <DictionaryEntryName>HakemuksenJattotapa.Teksti</DictionaryEntryName>
      <EntryType>BBIE</EntryType>
      <ObjectClassTerm>Toimeentulotukihakemus</ObjectClassTerm>
      <RepresentationTerm>Teksti</RepresentationTerm>
      <MinOccurence>0</MinOccurence>
      <MaxOccurence>1</MaxOccurence>
      <PropertyTerm>HakemuksenJattotapa</PropertyTerm>
      <XMLName>HakemuksenJattotapaTeksti</XMLName>
    </BusinessInformationEntity>
    <!-- Toimeentulotukihakemus jatkuu ... -->
  </BusinessDocument>
  <BusinessDocument>
    <DocumentName>Asiakaskertomus</DocumentName>
    <DocumentDefinition>Asiakirja sisältää asiakkaan palvelutapahtumat, yhteenvedon toimeentulotuen asiakas-
työstä, asiakaskäynneistä sekä raportoinnin asiakkuuden etenemisestä tehdyistä merkinnöistä ja syntyneistä
asiakirjoista. Asiakirja on luonteeltaan kronologisesti kumuloituva ja tapahtumatietoja kirjataan toimeentu-
lotukiprosessin eri vaiheissa.</DocumentDefinition>
    <!-- Asiakas, Laatija sekä Hakija kuten edellisessa asiakirjassa -->
    <BusinessInformationEntity>
      <DictionaryEntryName>Asiakaskertomus.Asiakaskertomus</DictionaryEntryName>
      <EntryType>ASBIE</EntryType>
      <ObjectClassTerm>ToimeentulotuenAsiakaskertomus</ObjectClassTerm>
      <PropertyTerm>Asiakaskertomus</PropertyTerm>
      <MinOccurence>0</MinOccurence>
      <MaxOccurence>1</MaxOccurence>
      <XMLName>Asiakaskertomus</XMLName>
    </BusinessInformationEntity>
    <BusinessInformationEntity>
      <DictionaryEntryName>Asiakaskertomus</DictionaryEntryName>
      <EntryType>ABIE</EntryType>
      <ObjectClassTerm>ToimeentulotuenAsiakaskertomus</ObjectClassTerm>
      <XMLName>Asiakaskertomus</XMLName>
    </BusinessInformationEntity>
    <!-- Asiakaskertomus jatkuu ... -->
  </BusinessDocument>
</BusinessDocumentLibrary>
```

G Teknisen asiakirjarakenteen XSLT-muunnostiedosto

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fn="http://www.w3.org/2005/xpath-functions" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:ccts="http://tikesos.fi/ccts#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:ak="http://sosmeta.fi/documents/XYZ#" xml:base="http://sosmeta.fi/documents/XYZ#">
  <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
  <xsl:template match="text()"/>
  <xsl:template match="/">
    <xsl:for-each select="//BusinessDocument">
      <xsl:variable name="DocumentName" select="DocumentName"/>
      <xsl:variable name="filename" select="concat($DocumentName,'.xml')"/>
      <xsl:value-of select="$filename"/>
      <xsl:result-document href="{$filename}">
        <xsl:for-each-group select="BusinessInformationEntity" group-adjacent="ObjectClassTerm">
          <xsl:variable name="groupKey" select="current-grouping-key()"/>
          <xsl:variable name="ObjectClassTermQualifiers" select="ObjectClassTermQualifiers"/>
          <xsl:variable name="QualifiedObjectClassTerm" se-
lect="concat($ObjectClassTermQualifiers,$groupKey)"/>
          <xsl:for-each select="current-group()">
            <xsl:variable name="EntryType" select="EntryType"/>
            <xsl:variable name="XMLName" select="XMLName"/>
            <xsl:variable name="PropertyXMLName" select="concat(lower-case(substring($XMLName, 1, 1)), sub-
string($XMLName, 2))"/>
            <xsl:variable name="PropertyTerm" select="PropertyTerm"/>
            <xsl:variable name="MinOccurence" select="MinOccurence"/>
            <xsl:variable name="MaxOccurence" select="MaxOccurence"/>
            <xsl:variable name="Label" select="Label"/>
            <xsl:variable name="Definition" select="Definition"/>
            <xsl:variable name="Notice" select="Notice"/>
            <xsl:variable name="DictionaryEntryName">
              <xsl:value-of select="DictionaryEntryName"/>
            </xsl:variable>
            <xsl:variable name="Example" select="Example"/>
            <xsl:variable name="PropertyTermQualifiers" select="PropertyTermQualifiers"/>
            <xsl:choose>
              <xsl:when test="$EntryType='ABIE'">
                <rdf:Description rdf:about="#{if($ObjectClassTermQualifiers) then
concat($ObjectClassTermQualifiers,$groupKey) else current-grouping-key()}">
                  <xsl:for-each select="current-group()">
                    <xsl:variable name="EntryType" select="EntryType"/>
                    <xsl:variable name="XMLName" select="XMLName"/>
                    <xsl:variable name="Definition" select="Definition"/>
                    <xsl:variable name="Example" select="Example"/>
                    <xsl:variable name="Group" select="Group"/>
                    <xsl:variable name="Domain" select="Domain"/>
                    <xsl:variable name="Notice" select="Notice"/>
                    <xsl:variable name="Label" select="Label"/>
                    <xsl:variable name="PropertyTermQualifiers" select="PropertyTermQualifiers"/>
                    <xsl:choose>
                      <xsl:when test="$EntryType='ABIE'">
                        <rdfs:subClassOf rdf:resource="#ABIE"/>
                        <rdf:type rdf:resource="http://tikesos.fi/ccts#{$EntryType}"/>
                        <ccts:hasNext rdf:resource="#{following-
sibling::BusinessInformationEntity[1]/DictionaryEntryName}"/>
                        <xsl:if test="$Definition">
                          <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                            <xsl:value-of select="$Definition"/>
                          </ccts:definition>
                        </xsl:if>
                        <xsl:if test="$Example">
                          <ccts:example rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                            <xsl:value-of select="$Example"/>
                          </ccts:example>
                        </xsl:if>
                        <xsl:if test="$Group">
                          <ccts:group rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                            <xsl:value-of select="$Group"/>
                          </ccts:group>
                        </xsl:if>
                        <xsl:if test="$Domain">
                          <ccts:domain rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                            <xsl:value-of select="$Domain"/>
```

```
                        </ccts:domain>
                      </xsl:if>
                      <xsl:if test="$Notice">
                        <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                          <xsl:value-of select="$Notice"/>
                        </ccts:notice>
                      </xsl:if>
                      <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$Label"/>
                      </rdfs:label>
                      <ccts:hasDictionaryEntryName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                        <xsl:value-of select="$DictionaryEntryName"/>
                      </ccts:hasDictionaryEntryName>
                      <ccts:hasObjectClassTerm rdf:resource="#{$groupKey}"/>
                      <xsl:if test="$ObjectClassTermQualifiers">
                        <ccts:hasObjectClassTermQualifiers
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                          <xsl:value-of select="$ObjectClassTermQualifiers"/>
                        </ccts:hasObjectClassTermQualifiers>
                      </xsl:if>
                    </xsl:when>
                    <xsl:otherwise>
                      <rdfs:subClassOf
rdf:nodeID="ID{concat($PropertyTermQualifiers,$groupKey,$XMLName)}"/>
                    </xsl:otherwise>
                  </xsl:choose>
                </xsl:for-each>
              </rdf:Description>
            </xsl:when>
            <xsl:otherwise>
              <rdf:Description rdf:about="#{$DictionaryEntryName}">
                <rdf:type rdf:resource="http://tikesos.fi/ccts#{$EntryType}"/>
                <xsl:if test="following-sibling::BusinessInformationEntity[1]/ObjectClassTerm=$groupKey">
                  <ccts:hasNext rdf:resource="#{following-
sibling::BusinessInformationEntity[1]/DictionaryEntryName}"/>
                </xsl:if>
                <xsl:if test="$Definition">
                  <ccts:definition rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                    <xsl:value-of select="$Definition"/>
                  </ccts:definition>
                </xsl:if>
                <xsl:if test="$Example">
                  <ccts:example rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                    <xsl:value-of select="$Example"/>
                  </ccts:example>
                </xsl:if>
                <xsl:if test="$Notice">
                  <ccts:notice rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                    <xsl:value-of select="$Notice"/>
                  </ccts:notice>
                </xsl:if>
                <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$Label"/>
                </rdfs:label>
                <ccts:hasDictionaryEntryName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$DictionaryEntryName"/>
                </ccts:hasDictionaryEntryName>
                <ccts:hasObjectClassTerm rdf:resource="#{$groupKey}"/>
                <xsl:if test="$ObjectClassTermQualifiers">
                  <ccts:hasObjectClassTermQualifiers
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                    <xsl:value-of select="$ObjectClassTermQualifiers"/>
                  </ccts:hasObjectClassTermQualifiers>
                </xsl:if>
                <ccts:hasPropertyTerm rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$PropertyTerm"/>
                </ccts:hasPropertyTerm>
                <ccts:minOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
                  <xsl:value-of select="$MinOccurence"/>
                </ccts:minOccurs>
                <ccts:maxOccurs rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                  <xsl:value-of select="$MaxOccurence"/>
                </ccts:maxOccurs>
                <xsl:choose>
                  <xsl:when test="EntryType='BBIE'">
                    <rdfs:subClassOf rdf:resource="#BBIE"/>
                    <xsl:variable name="RepresentationTerm" select="RepresentationTerm"/>
```

```
                    <ccts:onProperty rdf:resource="#{$PropertyXMLName}"/>
                    <ccts:hasRepresentationTerm rdf:resource="http://jhs-
suositukset.fi#{$RepresentationTerm}"/>
                  </xsl:when>
                  <xsl:when test="$EntryType='ASBIE'">
                    <rdfs:subClassOf rdf:resource="#ASBIE"/>
                    <xsl:variable name="AssociatedObjectClassTerm" select="AssociatedObjectClassTerm"/>
                    <xsl:variable name="PropertyTermQualifiers" select="PropertyTermQualifiers"/>
                    <xsl:variable name="QualifiedAssociatedObjectClassTerm" se-
lect="concat($PropertyTermQualifiers,$AssociatedObjectClassTerm)"/>
                    <ccts:onProperty rdf:resource="#{$PropertyXMLName}"/>
                    <xsl:variable name="Namespace" select="Namespace"/>
                    <ccts:hasAssociatedObjectClass rdf:resource="{if($Namespace) then
concat($StaticNamespace,$Namespace,'#',$QualifiedAssociatedObjectClassTerm) else
concat('#',$QualifiedAssociatedObjectClassTerm)}"/>
                  </xsl:when>
                </xsl:choose>
              </rdf:Description>
            </xsl:otherwise>
          </xsl:choose>
          <xsl:choose>
            <xsl:when test="$EntryType='ASBIE'">
              <xsl:variable name="AssociatedObjectClassTerm" select="AssociatedObjectClassTerm"/>
              <xsl:variable name="PropertyTermQualifiers" select="PropertyTermQualifiers"/>
              <xsl:variable name="QualifiedAssociatedObjectClassTerm" se-
lect="concat($PropertyTermQualifiers,$AssociatedObjectClassTerm)"/>
              <rdf:Description rdf:nodeID="ID{concat($PropertyTermQualifiers,$groupKey,$XMLName)}">
                <xsl:choose>
                  <xsl:when test="$MinOccurence=1">
                    <xsl:choose>
                      <xsl:when test="$MaxOccurence=1">
                        <owl:qualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:qualifiedCardinality>
                      </xsl:when>
                      <xsl:otherwise>
                        <owl:minQualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minQualifiedCardinality>
                      </xsl:otherwise>
                    </xsl:choose>
                  </xsl:when>
                  <xsl:otherwise>
                    <xsl:choose>
                      <xsl:when test="$MaxOccurence=1">
                        <owl:maxQualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxQualifiedCardinality>
                      </xsl:when>
                      <xsl:otherwise>
                        <owl:minQualifiedCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minQualifiedCardinality>
                      </xsl:otherwise>
                    </xsl:choose>
                  </xsl:otherwise>
                </xsl:choose>
                <owl:onProperty rdf:resource="#{$PropertyXMLName}"/>
                <xsl:variable name="Namespace" select="Namespace"/>
                <owl:onClass rdf:resource="{if($Namespace) then
concat($StaticNamespace,$Namespace,'#',$QualifiedAssociatedObjectClassTerm) else
concat('#',$QualifiedAssociatedObjectClassTerm)}"/>
                <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
              </rdf:Description>
              <xsl:if test="not(preceding-sibling::BusinessInformationEntity[XMLName=$XMLName])">
                <rdf:Description rdf:about="#{$PropertyXMLName}">
                  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                    <xsl:value-of select="$Label"/>
                  </rdfs:label>
                  <rdf:type rdf:resource="http://tikesos.fi/ccts#ASBIEProperty"/>
                </rdf:Description>
              </xsl:if>
            </xsl:when>
            <xsl:when test="$EntryType='BBIE'">
              <rdf:Description rdf:nodeID="ID{concat($PropertyTermQualifiers,$groupKey,$XMLName)}">
                <xsl:choose>
                  <xsl:when test="$MinOccurence=1">
                    <xsl:choose>
                      <xsl:when test="$MaxOccurence=1">
                        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:cardinality>
```

```
                        </xsl:when>
                        <xsl:otherwise>
                          <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minCardinality>
                        </xsl:otherwise>
                      </xsl:choose>
                    </xsl:when>
                    <xsl:otherwise>
                      <xsl:choose>
                        <xsl:when test="$MaxOccurence=1">
                          <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:maxCardinality>
                        </xsl:when>
                        <xsl:otherwise>
                          <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
                        </xsl:otherwise>
                      </xsl:choose>
                    </xsl:otherwise>
                  </xsl:choose>
                  <owl:onProperty rdf:resource="#{$PropertyXMLName}"/>
                  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Restriction"/>
                </rdf:Description>
                <xsl:if test="not(preceding-sibling::BusinessInformationEntity[XMLName=$XMLName])">
                  <xsl:variable name="RepresentationTerm" select="RepresentationTerm"/>
                  <!-- Property Identifier -->
                  <rdf:Description rdf:about="#{$PropertyXMLName}">
                    <xsl:choose>
                      <xsl:when test="$RepresentationTerm='Teksti'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Numero'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Nimi'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Pvm'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Hetki'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Kytkin'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Lkm'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Maara'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Aika'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#time"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Prosentti'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Arvo'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#decimal"/>
                      </xsl:when>
                      <xsl:when test="$RepresentationTerm='Koodi'">
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                      </xsl:when>
                      <xsl:otherwise>
                        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
                      </xsl:otherwise>
                    </xsl:choose>
                    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                      <xsl:value-of select="$Label"/>
                    </rdfs:label>
                    <rdf:type rdf:resource="http://tikesos.fi/ccts#BBIEProperty"/>
                  </rdf:Description>
                </xsl:if>
              </xsl:when>
            </xsl:choose>
          </xsl:for-each>
```

```
        </xsl:for-each-group>
      </xsl:result-document>
    </xsl:for-each>
  </xsl:template>
</xsl:style
```

H XHTML+RDFa asiakirjan esimerkkipohja

```
<?xml version="1.0" encoding="UTF-8"?>
<html version="XHTML+RDFa 1.0" xml:lang="fi" xmlns="http://www.w3.org/1999/xhtml" >
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>Esimerkkiasiakirja</title>
      <style type="text/css">
    body { width:210mm;}
    div { border: #000 0px solid; font-family: arial, times, serif; font-size:10pt; }
    div.document { margin-top:10mm; margin-left:20mm; margin-right: 7.12mm; width: 182.9mm; }
    div.main-holder { padding-top: 10mm; width: 182.9mm; float: left; }
    div.main-wrapper { width: 182.9mm; float: left;}
    div.meta { display: inline; margin: 0; padding: 0;}
    div.logo {
background:url('...')
    top left no-repeat;
    position:relative;
    width: 22.86mm;
    height: 100%;
    float:left; }
    div.header { height: 20mm;  float: left; border-bottom: #000 2px solid;
                 width: 182.9mm; word-wrap:break-word;}
    div.hdr1 { position:relative; width: 68.56mm; float:left;}
    div.hdr2 { position:relative; width: 68.56mm; float:left;}
    div.hdr3 { position:relative; width: 22.88mm; text-align: right; float:left;}
    div.header h1 { font-size: 10pt; font-weight: bold; margin: 0 0 1mm 0; padding: 0 0 0 0; }
    div.header h2 { font-size: 10pt; font-weight: normal; margin: 0 0 1mm 0; padding: 0 0 0 0; }
    div.list{ float: left; width: 182.9mm; }
    div.item {float: left; width: 182.9mm; margin-bottom: 5px; }
    div.title{  width: 50.9mm; float: left; word-wrap:break-word; }
    div.title h1 { margin: 0 0 0 0; font-size: 11pt;  font-weight: normal; }
    div.title h2 { margin: 0 0 0 0; padding-left: 4mm; font-size:10pt;  font-weight: normal;  }
    div.title h3 { margin: 0 0 0 0; padding-left: 8mm; font-size:9pt;  font-weight: normal;  }
    div.fulltitle { width: 182.9mm; float: left; margin-top:4.23mm;
                    margin-bottom: 2mm; word-wrap:break-word; }
    div.fulltitle h1 { width: 182.9mm; margin: 0 0 5mm 0; font-size: 11pt;
                       font-weight: bold; float: left;   }
    div.fulltitle h2 { width: 182.9mm; margin: 0 0 0 0; font-size: 11pt;
                       font-weight: bold; float: left;   }
    div.content{width: 132mm; float: right; }
    div.content p { margin: 0 0 0 0; font-size: 11pt; font-weight: normal; }
    div.fullcontent{width: 132mm; margin-left: 55mm; float: right; }
    div.content table { width: 132mm; }
    div.fullcontent table { width: 182.9mm; }
    table.table-item { border-collapse: collapse; }
    table.table-item td { width: 20%;  }
    table.table-item tr { margin-bottom: 5mm;  }
    td.title1 { margin-left: 3mm; float:left; }
    div.footer { float: left; height: 20mm; padding-top: 1mm; margin-top: 5mm;
                 border-top: #000 2px solid; width: 182.9mm; }
    div.ftr1 { position:relative; height: 100%; width: 60.96mm; float:left;}
    div.ftr2 { position:relative; height: 100%; width: 60.96mm; float:left;}
    div.ftr3 { position:relative; height: 100%; width: 60.96mm; float:left;}
    div.footer h1 { font-size: 10pt; font-weight: bold; padding: 0 0 0 0; }
    div.footer h2 { font-size: 10pt; font-weight: normal; padding: 0 0 0 0; }
    </style>
  </head>
  <body>
    <div class="document">
      <div class="header">
        <div class="logo"></div>
        <div class="hdr1"><h1>Organisaatio</h1><h2>Yksikkö</h2></div>
        <div class="hdr2"><h1>Asiakirjan tyyppi</h1><h2>Palvelutehtävä</h2></div>
        <div class="hdr3"><h2>31.12.2011</h2><h2>1 (1)</h2></div>
      </div>
      <div class="main-wrapper">
        <div class="main-holder">
            <div class="list">
              <div class="item" rel="sos:asiakasYksityishenkilo">
                <div class="title"><h1>Asiakas</h1></div>
                <div class="content">
                  <p><span property="sos:etuNimi">Essi</span> <span proper-
ty="sos:sukuNimi">Esimerkki</span></p>
                </div>
                <div class="content" rel="sos:yhteystiedot">
```

```
                <p><span property="sos:puhelinNumero">0123456</span><br/></p>
                <div rel="sos:osoite">
                  <p><span property="sos:osoiteTeksti">Esimerkkikatu 1 1234 Esimerkkilä</span></p>
                </div>
                </div>
              </div>
            <div class="item">
              <div class="fulltitle"><h1>Asiakirjan otsikko</h1></div>
            </div>
            <div class="item">
              <div class="title"><h1>1 Taso</h1></div>
              <div class="content"><p>Sisältöä</p></div>
            </div>
            <div class="item">
              <div class="title"><h2>2 Taso</h2></div>
              <div class="content"><p>Sisältöä</p></div>
            </div>
            <div class="item">
              <div class="title"><h3>3 Taso</h3></div>
              <div class="content"><p>Sisältöä</p></div>
            </div>
          </div>
        </div>
      </div>
      <div class="footer">
        <div class="ftr1"><h1>Postiosoite</h1></div>
        <div class="ftr2"><h1>Käyntiosoite</h1></div>
        <div class="ftr3"><h1>Yhteystiedot</h1></div>
      </div>
    </div>
  </body>
</html>
```